Classification by Frequent Association Rules

Md Rayhan Kabir
University of Alberta, Amii, Canada
mdrayha1@ualberta.ca

Osmar R. Zaïane
University of Alberta, Amii, Canada
zaiane@cs.ualberta.ca

ABSTRACT

Over the last two decades, Associative Classifiers have shown competitive performance in the task of predicting class labels. Along with the performance in accuracy, associative classifiers produce human-readable predictive rules which is very helpful to understand the decision process of the classifiers. Associative classifiers from early days suffer from the limitation requiring proper threshold value setting which is dataset-specific. Recently some studies eliminated that limitation by producing statistically significant rules. Though recent models showed very competitive performance with state-of-the-art classifiers, their performance is still impacted if the feature vector of the training data is very large. An ensemble model can solve this issue by training each base learner with a subset of the feature vector. In this study, we propose an ensemble model Classification by Frequent Association Rules (CFAR) using associative classifiers as base learners. In our approach, instead of using classical ensemble and a voting method, we rank the generated rules based on frequency and select a subset of the rules for predicting class labels. We use 10 datasets from the UCI repository to evaluate the performance of the proposed model. Our ensemble approach CFAR eliminates the limitation of high memory requirement and runtime of recent associative classifiers if training datasets have large feature vectors. Among the dataset we used, along with increasing accuracy in most cases, CFAR removes the noisy rules which enhances the interpretability of the model.

CCS CONCEPTS

• **Computing methodologies** → **Ensemble methods**; **Rule learning**;

KEYWORDS

Associative classifier, Ensemble Learning, Explainable Ensemble

ACM Reference Format:
Md Rayhan Kabir and Osmar R. Zaïane. 2023. Classification by Frequent Association Rules. In *Proceedings of ACM SAC Conference (SAC'23).* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/xx.xxx/xxx_x

1 INTRODUCTION

With the steady increase in use of machine learning in real-life scenarios, the importance of classification is de facts increasing. Classification is supervised machine learning that labels the data in different distinct class labels. In the last few decades, myriads classification algorithms have been developed. Among the classification algorithms, associative classifier is quite little known. Most state-of-the-art classification algorithms work in a black box fashion, that is, the decision process of the model remains opaque. For example, the performance of deep learning model is very good for many classification tasks but the model reveals far less information about the learned parameters and the decision process. Even though it is possible to get the learned edge weight and the node bias, they are not interpretable. This becomes troublesome in applications where it is important to know and understand the decision process, like medical diagnosis, financial decision making, etc. As machine learning is becoming popular, the explanation of the decision process of the machine learning model has also become a point of interest and with this, the importance of the rule-based classifier is returning. Associative classifiers are one of those rule-based classifiers which use association rule mining techniques [1] to discover frequent patterns in the training data. From these rules, class association rules are derived and used for the classification task. The test data is labeled with the class association rules that are generated during the training phase. A class association rule is in the form of $X \rightarrow Y$ where the antecedent X is a conjunction of co-occurring features and the consequent Y is the associated class label. Several associative classifiers have been proposed namely CBA [2], ARC [3], CMAR [4], CPAR [5]. One big advantage of associative classifiers is the human readability of the rules which are generated and used for predicting the class label. Thus the decision process becomes easier to understand. Though the associative classifiers show competitive performance against state-of-the-art classifiers and have interpretability, they suffer from the limitation of requiring threshold values, namely support and confidence, which are dataset-specific.

To solve this limitation, Li and Zaiane [7] proposed Sigdirect where they used the Kingfisher algorithm [8] to find statistically significant classification rules. Sood and Zaiane [9] showed SigDirect produces noisy rules which impact the performance of the predictive model. Thus, they proposed SigD2, a classifier based on SigDirect with a two-stage rule pruning strategy that removes the noisy rules, The improvement is promising as it reduces the number of rules without compromising the accuracy. It was shown that SigD2 outperforms other rule-based classifiers as well as classical classifiers such as SVM, Bayesian, C4.5, and even simple neural networks. However, the performance of both SigDirect and SigD2 is limited in terms of required memory and runtime if the feature vector of the feature space is large. An ensemble of classifiers each trained on a subset of the feature space could address this issue. Random forest[11] is a famous ensemble model which uses decision trees as base learners and showed competitive results over the years. Recently, Welke et al. [12] proposed Decision Snippet

SAC'23, March 27 –April 2, 2023, Tallinn, Estonia
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-9517-5/23/03…$15.00
https://doi.org/xx.xxx/xxx_x

Features where instead of classical max-vote strategy, they mined small frequent branches as decision snippets from the learned decision trees of the ensemble. They showed a linear model on top of the snippets provides competitive results and reduces the model size.

In this paper, we propose Classification by Frequent Association Rules (CFAR), to improve the performance of SigD2 in terms of memory requirement and run time for datasets having a large number of features without affecting the accuracy. As a base learner, we consider SigD2 as Sood and Zaiane [9] showed SigD2 outperforms other associative classifiers. In our model, instead of using the classical ensemble approach that is training each base learner with a subset of the features and predicting the final class by the max-vote of the base learners, we aggregate all the rules learned by the base learners and rank the rules based on their frequency of appearance among the base learners. We then predict the class label using the frequent rules. We start predicting with the most frequent rules and gradually add more rules which are less frequent. We keep adding rules while there is an increase in the accuracy. If after adding some less frequent rules, the accuracy does not increase, we stop adding rules and the highest accuracy till this step is reported as final accuracy. The main contributions of our work are:

(1) Our proposed model CFAR shows better performance in terms of accuracy compared to SigD2, classical ensemble, C4.5, Random forest and the decision snippet features (DSF) algorithm.
(2) CFAR significantly lessens the limitation of SigD2 of high memory requirement and long runtime for dataset with large feature vector space.
(3) CFAR uses frequent rules from the base learners which helps to remove noisy and less important rules. Thus with less rules the ensemble becomes more interpretable.

The following sections are organized as follows: Section 2 provides the necessary background and related works, Section 3 describes the methodology that we follow in our experiments, Section 4 is the evaluation and analysis of the performance of CFAR, and finally, Section 5 is the conclusion with some future research direction.

2 RELATED WORK

Though little known, many researchers worked extensively in the last few decades on associative classifiers. Apriori algorithm proposed by Agrwal and Srikant[1] first introduced the association rule mining technique. Further Liu et al. [2] proposed Classification By Association(CBA) where they use the association rule mining technique for the classification task. They used an apriori-based approach to mine class association rules from the training data and ranked them based on metrics and the rule with the highest rank is used for the classification task. In CBA, many noisy rules are generated which affects the performance of the model. Thus in their subsequent work [13], they attempt to remove the noisy rules. This approach improves the performance of the classifier by choosing the most accurate class association rules for the classification task. The success of using association rule mining in classification tasks grabbed the attention of many researchers. Several initiatives were taken to improve the performance of associative classifiers. Classification based on Predictive Associative Rules (CPAR)[5] was proposed by Yin and Han. CPAR uses a greedy approach to generate association rules from the training data and evaluate each of the rules by an expected accuracy. Finally, best k rules are selected from the generated rules for the classification task. Another approach to improve the classification by association rule is CMAR (Classification based on Multiple Association Rules) [4] which extends the FP-growth [6] method to build an FP-tree for class distribution association. In this approach, the class association rules are stored in a special data structure. The generated rules are pruned based on confidence, correlation, and database coverage. Using multiple strong association rules with a Weighted χ^2 measure, a datapoint is labeled to the appropriate class.

Antonie and Zaiane [3] proposed another Association rule-based classifier that they apply for text categorization. They put forward two different approaches; one, ARC-AC, considering all generated rules from the whole training set, and one, ARC-BC, where data from each class is mined separately allowing the handling of unbalanced datasets. Another approach, CCCS, proposed by Arunasalam and Chawla [14] introduces a measure named "Complement class support" (CCS). The authors claim CCS guarantees a positive correlation between the class label and the generated rules. Antonie et al. [15] propose a two-stage associative classifier where associative classification rules are discovered in the first stage and in a second stage, another algorithm learns how to use those rules for class prediction. Instead of basing the selection of rules to apply during inference on some heuristics, they use a neural network to learn how to predict the best rules to apply. This approach showed improved efficiency in terms of accuracy.

One limitation of associative classifiers is the rule generation and the need to evaluate a huge number of rules as there are many noisy rules among them. To overcome this problem Zaiane and Antonie [16] performed an extensive study with the focus of reducing the number of rules without decreasing the accuracy of the classifier. The authors propose a new pruning strategy to reduce the number of class association rules. They also propose different heuristics for selecting rules which can obtain high accuracy for a given instance. However, it remains that the proper support and confidence values have to be selected and tuned. This is one major drawback of associative classifiers inherited from association rule mining. The performance of the model largely depends on these values. It is a tedious task to find the proper confidence and support values for each of the datasets, hampering the adoption of associative classifiers. To solve this issue, Li and Zaiane [7] propose SigDirect where they improvised the Kingfisher Algorithm [8] to find the statistically significant rules for classification, instead of frequent ones. Their proposed method increases the accuracy of the classifier. The main contribution of their work is eliminating the necessity of annoying support and confidence thresholds. However, SigDirect also has some limitations. Sood and Zaiane [9] showed noisy rules can be further eliminated by proposing SigD2, a two-stage pruning technique that can reduce the number of rules without jeopardizing the accuracy of the classifier and therefore making a learned model even more practically interpretable.

As we stated earlier, SigDirect and SigD2 suffer from the limitation of high memory requirement and long run time if the feature vector of the dataset is large, which can be solved by an ensemble model where each base learner is trained on a subset of the feature vector. In an extensive study of ensemble models, Bauer and Kohavi [17], empirically showed the ensemble models can enhance the performance of a classifier for most of the classification tasks. But the classical ensemble models follow the process of training base learners and taking the vote of the base learners to predict the final class label. Random forest [11] also follows this architecture but this affects the interpretability of the model. In recent work, Welke et al. [12] showed instead of following classical ensemble architecture, selecting frequent branches from the base learners and making another simple model using those branches can be efficient both in terms of accuracy and interpretability. Thus in this paper, we propose CFAR where we first train each of the base learners with a subset of the feature vector and generate class association rules. We then aggregate all the generated rules and rank them based on their frequency among the base learners, select rules with high frequency, and make the final prediction using the selected rules.

3 METHODOLOGY

In this section we briefly describe our proposed architecture for the ensemble model. In the architecture we use SigD2 as base learner because from the literature we found SigD2 outperforms other associative classifier in terms of accuracy and number of generated rules [9]. In our approach at first we tried with classical ensemble approach. In most of the cases of ensemble, each of the base learner is trained on a subset of instances. But in our case, we train our base learners with a subset of the feature vector and all instances, as one of the limitation of SigD2 is its performance with a large feature vector size. At first, we experiment with classical ensemble with random subsampling method. We provide details of this procedure in the next subsection.

3.1 Classical ensemble approach

In this approach at first, we train 100 base learners each with a random subsample of size N of the original feature space. We provide the process of creating a random subsample in Algorithm 1.

Algorithm 1 Subsample generation by randomly selecting features

```
Input: features: all features of the feature space; N: Number of
features in each subsample
Output: 100 subsamples of the feature space
 1: all_subsamples ⟵ []
 2: for i in range(100) do:
 3:     new_subsample ⟵ []
 4:     n ⟵ 0
 5:     while n < N
 6:         feature ⟵ randomly select a feature from features
 7:         new_subsample.append(feature)
 8:         $n \longleftarrow n + 1$
 9:     end while
10:     all_subsamples.append(new_subsample)
11: end for
12: return all_subsamples
```

This classical approach where base learners vote on the output class label has some limitations. Since the base learners are trained on different subsets a vote would be biased. Some features which are selected many times might have a bias on the prediction and affect the final result. Thus the performance might be hampered. For this reason, instead of taking the vote of the base learners, we propose CFAR where we take the rules learned by the base learners and rank them according to their frequency among base learners. In the next subsection, we provide a brief explanation of our proposed model.

3.2 Classification by Frequent Association Rules

One advantage of making an ensemble of Associative classifiers is that we can collect all the rules learned from the training data set. These rules are used to label test data to a specified class label. However, selecting all the rules might include noisy rules which might affect the performance of the model. To solve this we select the rules which enhance the performance of the model. For this, we introduce the term Relative Frequency Ratio(RFR). After collecting all the rules generated by the base learners, we count the frequency of the rules. We find the Rule having the maximum frequency that we note as *max_frequency*. We calculate the RFR of any rule R by Equation 1.

$$Relative\_frequency\_ratio(R) = \frac{Frequency(R)}{max\_frequency} \tag{1}$$

To predict the class label, we need to select rules from the generated rules. We consider a threshold T and select all the rules whose RFR is greater than or equal to T. The process of selecting rules is provided in Algorithm 2.

Algorithm 2 Rules selection based on RFR

```
Input: Rules: all rules generated by base learners; T: Threshold
value to select Rules.
Output: Selected_rules: rules with RFR ≥ T
1: Selected_rules ⟵ []
2: for r in Rules do:
3:     if RFR(r) ≥ T:
4:         Selected_rules.append(r)
5:     end if
6: end for
7: return Selected_rules
```

Unfortunately as depicted on figure 1, there is no optimum value for T with all datasets. Therefore we design our model in such a way that the model itself can find the value of T for which we can get maximum accuracy. To achieve this goal, we outline a 3 step ensemble classifier. We describe each of the steps below:

3.2.1 Rule generation. In the rule generation phase, we follow the random sub-sampling procedure and train 100 SigD2 base learners. We divide the dataset into train and validation data. We take 80% of the whole dataset as training data and the rest 20% as validation data. Then again we divide the training data as training and test data in the ratio of 80% and 20% respectively. To train each base learner we take a subset of the feature vector of size 30. Since the Flare dataset already has a feature vector of size 30, we take a subset of size 15 in the case of Flare. After training 100 base learners with

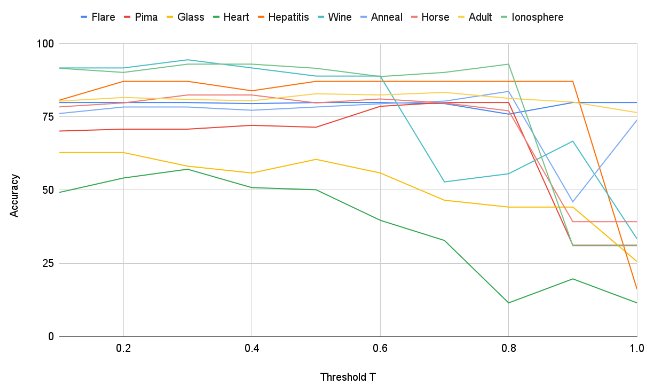

Figure 1: Performance of the model for different values of Threshold T on 10 different datasets

the training data, we gather all the rules generated by the base learners. We calculate the Relative Frequency Ratio (RFR) for each of the rules using Equation 1.

3.2.2 Find optimum value for T. To find the optimum value for T, we try with different values in a linear search fashion. We assume, the importance of a rule depends on its frequency among the base learners. A rule being more frequent defines more importance. Thus while experimenting with different values of T, at first we try with $T = 1$. We select rules from the generated rules using algorithm 2 and predict the class label with the selected rules. In the next step, we decrease the value of T by 0.1 and with that value we perform a rule selection and class prediction. In each step, we calculate the accuracy. We continue to decrease the value of T till we get the best improvement over the accuracy. If for a certain value of T, there is no improvement, we stop searching and fix the T value from the previous step as the optimum T value.

3.2.3 Prediction. The last step is the prediction. From the previous step we get the value of T which provides best result. We select rules using that value of T and predict the class label of the validation data. With the predicted class label, we calculate the performance of the model. The whole architecture is shown in Figure 2.

4 RESULT ANALYSIS

We use 10 different UCI datasets [18] to evaluate our proposed model CFAR. Before using a dataset we discretize the numerical values as stated in [19]. We convert the features to a binary feature vector. We used the same vector form of discretized values for all our experiments with all contenders. Thus the results of mentioned algorithms can be slightly different from their original papers. As we mentioned earlier, to test our model CFAR, we use 20% of the dataset to validate our model. We further divide the rest 80% of the data into train and test data in the ratio of 80% and 20% respectively. For all other models, we use 80% of the data to train the model and rest 20% is used as test data.

4.1 Performance evaluation

We compare our model with SigD2 as Sood and Zaiane [9] showed SigD2 outperforms other associative classifiers, rule-based classifiers, and other state-of-the-art machine learning models. To compare the result with another interpretable model we consider C4.5 proposed by Quinlan [10]. As we are making an ensemble architecture we considered the performance of random forest[11] which is an ensemble of decision trees. Our idea was motivated by the work of Decision snippet features (DSF) model[12] so we are also interested to compare the performance of our model with it. We also considered the classical ensemble architecture where we used 100 base learners each trained with a subset of the feature vector of size 30. The comparison in the accuracy of SigD2, C4.5, DSF model, Random forest, classical ensemble approach, and CFAR is provided in Table 1.

From Table 1, we can see in 5 among the 10 datasets CFAR has better performance than other classifiers and ensemble models. The random forest has better performance than CFAR in 3 datasets among which in Anneal dataset the margin is very large. In Glass dataset DSF performs better than CFAR and the margin is also very large. Another interesting observation in these 2 datasets that is in Anneal and Glass dataset classical ensemble approach has better performance than CFAR. That means in these two datasets the approach of CFAR where we select the rules based on the Relative Frequency Ratio (RFR) does not perform well. Further analysis is needed on these two dataset to understand why CFAR is not performing well. Only in one case, Flare, SigD2 performs better than all other models. On Average, though very close to DSF, CFAR has better performance than other models.

Algorithm	p-value
CFAR vs SigD2	0.0040
CFAR vs C4.5	0.0019
CFAR vs DSF	0.2303
CFAR vs Random Forest	0.0360
CFAR vs Classical Ensemble	0.0001

Table 2: Statistical result

We also perform a statistical test to understand whether the improvement in accuracy by CFAR is significant or not. For this, we use paired t-test from the work of Hsu and Lachenbruch [20]. In the test, our null hypothesis is the improvement in accuracy by CFAR is not significant. We repeated the whole experiment 20 times for each of the datasets and calculated the accuracy of each of the models. Then we calculated the difference in the accuracy for each pair of the model. With the difference, we calculated the p-value. This p-value signifies whether the difference is significant or not. If the p-value is less than the threshold value alpha(0.05) then we can reject the null hypothesis and say the difference in improvement by our proposed model CFAR is significant. We provide the calculated p-value in Table 2. From Table 2, we can say, except for DSF, in the dataset we used, CFAR shows significant improvement over accuracy.

Further, we analyze the precision and recall values of the models. Figure 3 and Figure 4 show the precision and recall values of the

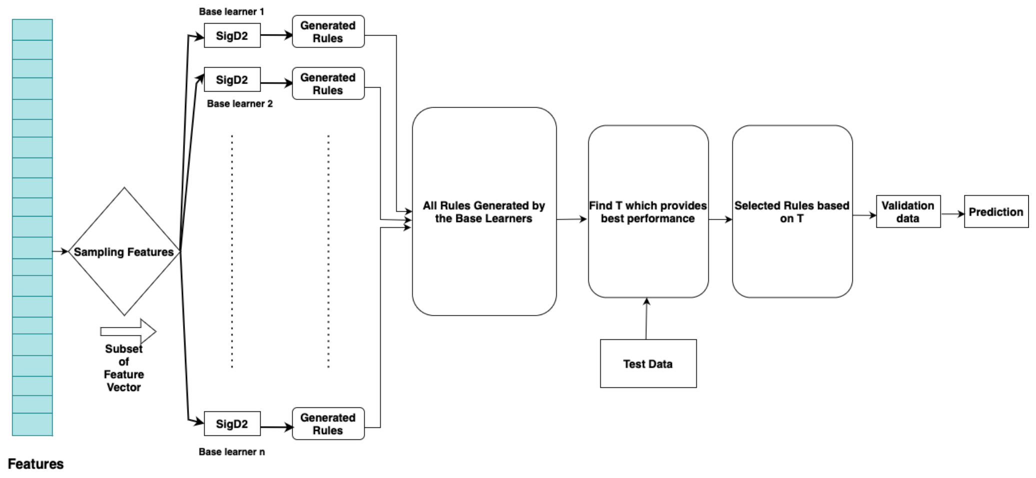

Figure 2: Proposed model of CFAR: We generate rules by base learners using training data. We apply generated rules on test data to find the T which provides best performance. With the help of T we select final rules and apply them on validation data to calculate performance of the model

Dataset	#cls	#record	Feature Vector Size	C4.5	RF	DSF	SigD2	CE	CFAR
Flare	9	1389	30	75.54	69.73	78.78	**84.39**	76.43	79.86
Pima	2	768	36	76.62	67.41	74.92	76.44	78.96	**79.22**
Glass	7	214	41	62.79	72.09	**72.42**	48.84	68.13	58.14
Heart	5	303	47	50.82	52.45	52.46	40.98	48.51	**57.38**
Hepatitis	2	155	54	70.97	78.38	74.19	81.54	76.21	**83.87**
Wine	3	178	75	91.67	78.78	92.44	92.7	81.29	**94.44**
Anneal	6	898	67	97.22	**98.89**	97.78	92.22	88.86	83.33
Horse	2	368	83	78.38	75.67	79.73	75.68	81.25	**83.78**
Adult	2	48842	95	84.06	**84.97**	82.53	84.59	81.81	83.31
Ionosphere	2	336	155	88.73	**95.78**	88.18	90.15	90.77	92.96
Average				77.68	77.42	79.34	76.75	77.22	**79.62**

Table 1: Accuracy of C4.5, Random Forest (RF), Decision Snippet Features (DSF), SigD2, Classical ensemble approach (CE) and CFAR

models respectively. From the two figures, we can see CFAR has a very competitive performance compared to the other models in terms of precision and recall.

4.2 Memory Requirement

One of our main goals for making an ensemble of SigD2 is to eliminate the limitation of requiring very high amount of memory with the increase of the feature vector size. We measure the memory required by the models using the python library *psutil*. Table 3 shows the memory requirement for SigD2, CFAR and other contenders.

From Table 3, we can see, in the comparison of memory requirement between CFAR and SigD2 has two different scenarios. When the size of the feature vector is small, the memory requirement of SigD2 and CFAR are very close. However, with the increase in feature vector size, the memory requirement of SigD2 increases drastically while for CFAR the memory requirement remains stable.

Dataset	C4.5	RF	DSF	SigD2	CFAR
Flare	107	132	113	106	**105**
Pima	103	128	112	**102**	105
Glass	109	128	112	110	**107**
Heart	**105**	129	112	107	119
Hepatitis	**103**	127	113	113	108
Wine	**104**	128	112	165	107
Anneal	**107**	129	113	150	108
Horse	109	128	112	235	**108**
Adult	**195**	253	196	257	255
Ionosphere	**124**	128	113	2519	136

Table 3: Comparison of Memory requirement(MB)

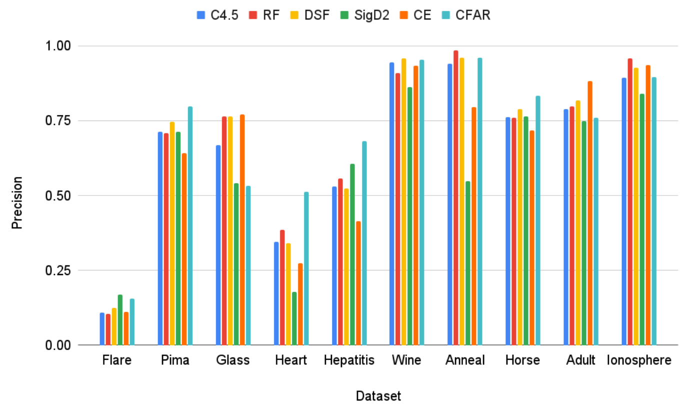

Figure 3: Precision of C4.5,Random Forest (RF), Decision Snippet Features (DSF), SigD2, Classical ensemble approach (CE) and CFAR

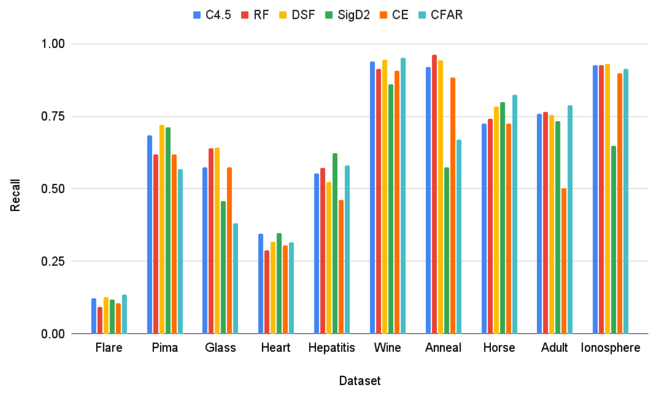

Figure 4: Recall of C4.5,Random Forest (RF), Decision Snippet Features (DSF), SigD2, Classical ensemble approach (CE) and CFAR

Thus we can say, CFAR eliminates the high memory requirement of SigD2 in case of a dataset with a large feature vector size.

4.3 Runtime

Another limitation of SigD2 is the runtime: it increases with the increase of feature vector space. The architecture of CFAR should also solve that issue as for each of the base learners, we are taking a fixed-size subset. To understand this, we also measured the runtime of CFAR and SigD2. The comparison is shown in Table 4. All the experiments are done on the machine with an Intel core i7 processor and 16 GB of RAM.

From Table 4, we can see that C4.5, random forest, and DSF model are faster than SigD2 and CFAR in classification tasks. This was expected. However, between SigD2 and CFAR, the dataset having a small size of feature vector, SigD2 is faster than CFAR. This is also expected since CFAR has 100 base learners which are trained

Dataset	C4.5	RF	DSF	SigD2	CFAR
Flare	1.72	0.85	0.17	**13.79**	27.79
Pima	1.76	0.15	0.10	**0.47**	28.32
Glass	0.32	0.12	0.32	**1.48**	26.46
Heart	0.69	0.13	0.34	**8.55**	67.51
Hepatitis	0.23	0.11	0.08	**10.41**	26.58
Wine	0.17	0.11	0.07	**0.46**	9.61
Anneal	2.10	0.14	0.10	**7.99**	23.75
Horse	0.92	0.13	0.09	32.49	**23.33**
Adult	13.20	3.19	1.60	263.12	**200.79**
Ionosphere	1.86	0.12	0.08	1905.44	**27.09**

Table 4: Comparison of Run time(seconds)

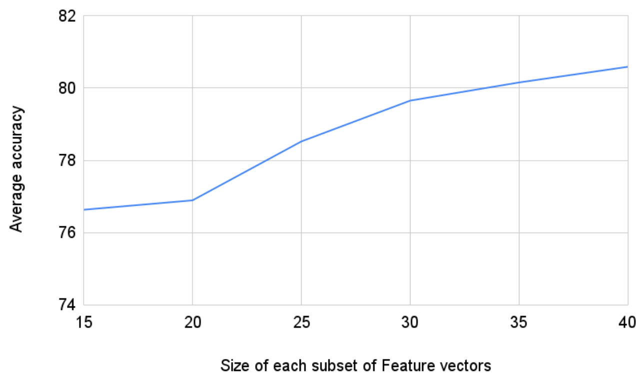

Figure 5: Average accuracy of 9 datasets for different size of the subset of Feature vector

sequentially. No parallel processing is performed at this stage in the experiment. Thus for datasets having small feature vector size, SigD2 is faster than CFAR. When the size of the feature vector is large, CFAR wins by a significant amount despite the overhead of the ensemble running sequentially.

4.4 Performance analysis on different sizes of the subset of feature vector

In our whole experiments with CFAR we always selected a subset of the feature vector of size 30. This is because from experiments, we found, the performance of SigD2 to be affected if the feature vector size is greater than 30. In this section, we conduct an experiment for a different size for the subset of feature vector. We start the experiment by selecting subset of size 15 and in each step, we increase the size by 5 until a size of 40. In Figure 5, Figure 6, and Figure 7 we can see the average accuracy, average memory requirement, and average runtime of the datasets for different sizes of subset of feature vector respectively. In this experiment, we exclude the Flare dataset as this dataset has a feature size less than the required feature size for this experiment. For the same reason, we also exclude the Pima dataset when we take a subset of feature vector with size 40.

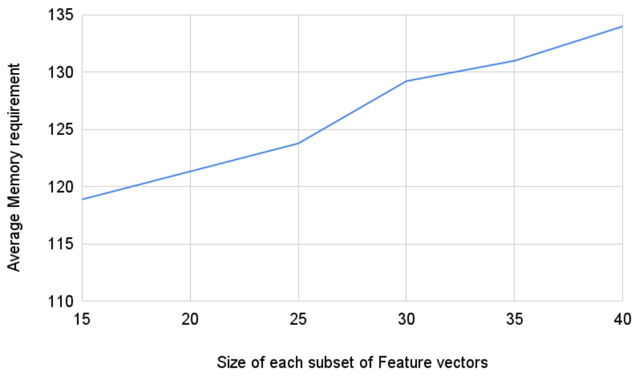

Figure 6: Average Memory requirement (MB) of 9 datasets for different size of subset of Feature vector

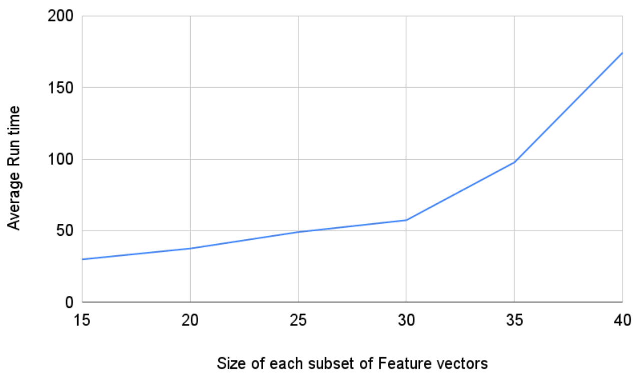

Figure 7: Average Runtime (Seconds) of 9 datasets for different size of subset of Feature vector

Figure 5 shows an increase of the accuracy with the increase of the size of the subset of feature vector. In the beginning, the rate of increase in accuracy is high but with the increase in the size of the subset, the rate of increase slows down. This shows that the more features we can bundle in a feature subspace for a base learner, the more CFAR can take advantage of possible feature inter-dependence. Figure 6 and 7 show the memory requirement and runtime increase with the increase of the size of the feature subspace. Beyond 30, this increase accelerates. Therefore we conclude that 30 is a good compromise for a good accurate while considering memory requirement and runtime.

4.5 Effect of Number of base learners

In our model, we use 100 base learners. We are interested to know the effect of the number of base learners in CFAR. Thus we tested the model with a different number of base learners. In Figure 8 we can see the performance of CFAR for a different number of base learners. In this experiment, we consider the size of the subset of the feature vector to be 30 except for Flare dataset where we use 15.

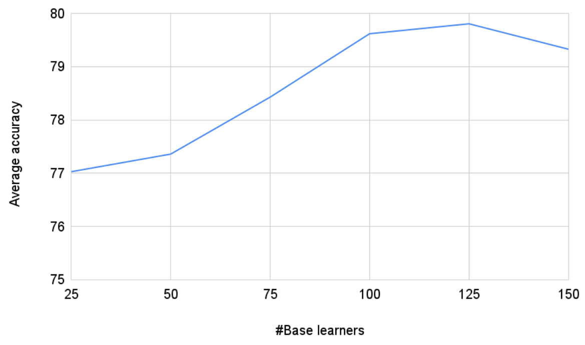

Figure 8: Effect of number of base learners on average accuracy of 10 datasets

From Figure 8, we can see when the number of base learners is reduced the average accuracy is low. With the increase of the base learners, accuracy increases. Beyond 100 base learners, the accuracy almost reaches a plateau then decreases. Considering the runtime there seems no advantage in having an ensemble larger than 100.

4.6 Interpretability and explainability

The major advantages of the associative classifier is the production of human-readable rules and by analyzing the rules it is easier to understand the decision process of the classifier. The associative classifier is therefore in itself explainable. An ensemble, however, is not straight forwardly explainable. In the classical ensemble approach, each of the base learners predicts the class label of test instances, and the final prediction is determined by a maximum vote. In the case of the ensemble of associative classifiers, to interpret the decision process we need to keep track of the base learners who voted for the final class label and then get the rules from those base learners. In this process, the number of base learners who voted for the final class label can be quite large and we need to consider the decision process of each of the base learners. Along with this, there can be some rules which have different weights in different base learners. This way of interpreting results can be difficult and time-consuming and create huge confusion.

With the CFAR approach, instead of using the max vote strategy, we collect all the rules and select rules based on their frequency. This process reduces the number of rules as well as eliminates the necessity of understanding the decision process of each of the base learners who voted for the final class label. In Table 5 we provide a comparative analysis of the number of total generated rules by the base learners, number of unique rules, and number of rules selected by CFAR for the classification process.

From Table 5 we can see, for each of the datasets, the total number of generated rules is very high compared to the selected rules. For example in Horse dataset, the total generated rules are 1848. But in the case of CFAR, only 13 rules are selected for the final prediction. Thus to understand decision process of the classifier we need to analyze only 13 rules. In Table 6, we provide the rules selected by CFAR for the final class prediction from the generated rules with

Dataset	#Total Rules	#Unique Rules	#Selected Rules
Pima	1462	427	8
Glass	2591	511	134
Heart	3121	1202	37
Hepatitis	1272	251	6
Wine	1339	142	33
Anneal	2258	464	19
Horse	1848	568	13
Adult	1946	160	66
Ionosphere	2118	634	78
Average	1993.11	484.33	43.78

Table 5: Number of rules at each stage of the CFAR

their frequency and Relative Frequency Ratio (RFR). From Table 6 we can see it is very convenient to understand the decision process of CFAR.

Selected Rules	Frequency	RFR
0 $\longrightarrow$ 0;(0.5277,0.873,-54.379)	40	1
63 $\longrightarrow$ 1;(0.0340,0.889,-6.068)	39	0.975
49 $\longrightarrow$ 0;(0.0894,0.840,-4.528)	33	0.825
1 $\longrightarrow$ 1;(0.3064,0.791,-57.129)	31	0.775
64 $\longrightarrow$ 0;(0.1447,0.944,-13.211)	26	0.65
55 $\longrightarrow$ 0;(0.1191,0.903,-8.601)	25	0.625
60 $\longrightarrow$ 0;(0.1915,0.738,-4.093)	25	0.625
15 $\longrightarrow$ 0;(0.4000,0.718,-8.093)	25	0.625
48 $\longrightarrow$ 0;(0.1957,0.821,-8.635)	24	0.6
54 $\longrightarrow$ 0;(0.1064,0.833,-5.093)	24	0.6
78 $\longrightarrow$ 0;(0.1447,0.919,-11.476)	24	0.6
62 $\longrightarrow$ 1;(0.0681,0.800,-9.304)	24	0.6
12 $\longrightarrow$ 1;(0.0936,0.595,-5.567)	24	0.6

Table 6: Ranked selected rules for final class prediction by CFAR with their count and RFR. Each rule is in the form of "Antecedent → Class label;(support, confidence, -ln(p-value))" where Antecedent is a conjunction of tokenized features. For interpretability, tokens are mapped back to features (attribute-value pairs)

5 CONCLUSION AND FUTURE WORKS

We propose an ensemble technique Classification by Frequent Association Rules (CFAR) using SigD2 as base learner where instead of the classical max voting strategy we aggregate all the rules generated by the base learners and select frequent rules for the classification task. An experiment with 10 different datasets shows CFAR increases the accuracy in most cases. CFAR also eliminates the limitation of high memory requirement and runtime of SigD2 in case of a dataset having a large feature vector size. In CFAR, instead of the classical ensemble process, we design the ensemble in such a way that the decision process remains human-readable and explainable, and can be understood by analyzing a very small number of rules.

Along with providing promising performance in terms of accuracy, memory requirement, and run time, CFAR also preserves interpretability. Understanding the decision process of CFAR is easier than any other ensemble model. In our model, we use SigD2 as base learner. We can also use other associative classifiers as base learners in this model. It would be interesting to use other associative classifiers as base learners and compare their performance. our study comprises applying CFAR only to tabular data. In the future, we want to deploy CFAR on text and image data. In our model, CFAR is a framework where we replaced the max voting strategy with aggregating the association rules for final prediction. It would be interesting to explore the same strategy in other interpretable ensemble models.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. "Fast algorithms for mining association rules." Proc. 20th int. conf. very large data bases, VLDB. Vol. 1215. 1994

[2] Bing Liu and Wynne Hsu and Yiming Ma. "Integrating classification and association rule mining." Kdd. Vol. 98. 1998.

[3] ML Antonie and Osmar R. Zaiane. "Text document categorization by term association." IEEE International Conference on Data Mining (ICDM). pp. 19–26 (2002)

[4] Wenmin Li and Jiawei Han and Jian Pei. "CMAR: Accurate and efficient classification based on multiple class-association rules." Proceedings 2001 IEEE international conference on data mining. IEEE, 2001.

[5] Xiaoxin Yin and Jiawei Han. "CPAR: Classification based on predictive association rules." Proceedings of the 2003 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2003.

[6] Jiawei Han, Jian Pei, and Yiwen Yin. "Mining frequent patterns without candidate generation." ACM sigmod record 29.2 (2000): 1-12.

[7] Jundong Li and Osmar R. Zaiane. "Exploiting statistically significant dependent rules for associative classification." Intelligent data analysis 21.5 (2017): 1155-1172

[8] Wilhelmiina Wilhelmiina and Matti Nykänen. "Efficient discovery of statistically significant association rules." 2008 Eighth IEEE international conference on data mining. IEEE, 2008.

[9] Nitakshi Sood and Osmar R. Zaiane. "Building a competitive associative classifier." International Conference on Big Data Analytics and Knowledge Discovery. Springer, Cham, 2020.

[10] J. Ross Quinlan. C4. 5: programs for machine learning. Elsevier, 2014.

[11] Leo Breiman. 2001. "Random Forests." Machine Learning 45: 5–32.

[12] Pascal Welke, Fouad Alkhoury, Christian Bauckhage, Stefan Wrobel. "Decision Snippet Features." 2021 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021.

[13] Bing Liu and Yiming Ma and Ching Kian Wong. "Improving an association rule based classifier." European Conference on Principles of Data Mining and Knowledge Discovery. Springer, Berlin, Heidelberg, 2000.

[14] Bavani Arunasalam and Sanjay Chawla. "CCCS: a top-down associative classifier for imbalanced class distribution." Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. 2006.

[15] ML Antonie and Osmar R. Zaiane and Robert C. Holte. "Learning to use a learned model: A two-stage approach to classification." Sixth International Conference on Data Mining (ICDM'06). IEEE, 2006.

[16] Osmar R. Zaiane and ML Antonie. "On pruning and tuning rules for associative classifiers." International Conference on Knowledge-Based and Intelligent Information and Engineering Systems. Springer, Berlin, Heidelberg, 2005

[17] Eric Bauer and Ron Kohavi. "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants." Machine learning 36.1 (1999): 105-139.

[18] Dheeru Dua and Casey Graff. "UCI machine learning repository." (2017)

[19] Frans Coenen, The lucs-kdd software library, 2004. [Online]. Available: https://cgi.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN/lucs-kdd_DN.html

[20] Henry Hsu, and Peter A. Lachenbruch. "Paired t test." Wiley StatsRef:statistics reference online (2014).