Stroke Prediction using Machine Learning Algorithms

Saghar Jiantavana, Fabliha Faiza Maliha
Ryerson University

Abstract

In this work, we consider the task of identifying stroke patient from patient's history. Predicting patient with a stroke can be simply considered as a binary classification problem where the algorithm predicts if a patient has stroke or not. We use the publicly available stroke prediction dataset [1] from kaggle[1]. We significantly pre-process the original dataset and sub-sample it as the dataset is highly unbalanced. After the preprocessing and sub-sampling steps we make sure the dataset contains reasonable number of instances for each class. Then we use eight different machine learning based algorithms, such as Naive Bayes, j48, Multi-layer perceptron, logistic regression, Voted Perceptron, Meta with AdaBoostM1, rule based oneR and IBK (k=10) for the stroke prediction task. Despite the imbalance issue in the dataset, all the learning algorithms achieve reasonable classification accuracy under different settings. We also perform extensive experiments with attribute selection where we only use a subset of attributes for the stroke classification task. We also identify which attributes have higher influence in the decision making process of the classification task. Interesting, the overall classification performance with attribute selection is competitive with the scenario where we use all the attributes. Note that all the experiments are performed in publicly available weka data mining tool.

1. Dataset Description and Pre-processing

We use the publicly available stroke prediction dataset from *kaggle*. This dataset contains total 5,110 instances including 4,861 no-stroke instances and 249 stroke instances. There are total 12 attributes in the dataset including id, gender, age, hypertension, heart-diseases, ever-married, work-type, residence-type, average glucose level, bmi, smoking status and stroke class. As the dataset is highly imbalanced, we perform few pre-processing steps before feeding the data to a learning algorithm as follows:

Dataset setting	Total	
	Stroke (1)	No Stroke (0)
Original dataset (w/o sub-sampling)	249	4861
New dataset (with sub-sampling)	249	981

Table 1. Details of the stroke dataset under two different settings.

- We used a publicly available code to transfer csv file to arrf file.
- The dataset has 201 instances with missing value for bmi. This accounts for 4% of our dataset .Since we are dealing with a small number of instances belonging to stroke class, we can not afford removing instances therefore we replaced them with mean/mode. In order to do so we identified and replaced these missing values using a Weka filter called ”ReplaceMissingValues”.
- We removed the ID attribute as it does not play an important role in the decision making process in the stroke classification task.
- We transfer the hypertension attribute values from numeric to nominal as the attribute has only two distinct values.
- We transfer the heart-disease attribute values from numeric to nominal as the attribute has only two distinct values, 0 indicates no heart disease while 1 represents the patient has heart disease.
- We transfer the bmi attribute from string to numeric values through manipulation of the arrf file headers. We replaced all the N/A values for this attribute with question marks through manipulation of the arrf file. Since question marks represent missing values in weka format , this could help us to apply the proper Weka filter and replace the missing values as mentioned above.
- Since our dataset is highly imbalanced, with 4861 no stroke and 249 stroke instances, it is clearly biased to-

[1] www.kaggle.com/fedesoriano/stroke-prediction-dataset

Attributes	Type	Missing	Unique	Distinct	Statistics (mean)	Statistics (stdDev)	Statistics (max)	Statistics (min)
ID	Numeric	0%	5110	5110	36517.8	21161	72940	67
age	Numeric	0%	0%	104	43.227	22.613	82	0.008
hypertension	Numeric	0%	0%	2	0.097	0.297	1	0
heart-disease	Numeric	0%	0%	2	0.054	0.226	1	0
average glucose level	Numeric	0%	3065	2979	106.148	45.284	271.74	55.12
bmi	Numeric	4%	73	418	28.893	7.854	97.6	10.3
class(stroke)	Numeric	0%	0%	2	0.049	0.215	1	0

Table 2. Details of the numeric attributes of the stroke dataset under different settings.

Attributes	Type	Missing	Unique	Distinct	Label	Count	Weight
gender	Nominal	0%	1(0%)	3	1. Male	2115	2115.0
gender	Nominal	0%	1(0%)	3	2. Female	2994	2994.0
gender	Nominal	0%	1(0%)	3	3. Other	1	1.0
ever married	Nominal	0%	0(0%)	2	1.Yes	3353	3353.0
ever married	Nominal	0%	0(0%)	2	2.No	1757	3353.0
work type	Nominal	0%	0%	5	1. Private	2925	2925.0
work type	Nominal	0%	0%	5	2. Self-employed	819	819.0
work type	Nominal	0%	0%	2	3.Govt job	657	657.0
work type	Nominal	0%	0%	2	4. Never Worked	22	22.0
Residence type	Nominal	0%	0%	2	1. Urban	2596	2596.0
Residence type	Nominal	0%	0%	2	1. Rural	2514	2514.0
smoking status	Nominal	0%	0%	4	1.Formerly Smoked	885	885.0
smoking status	Nominal	0%	0%	4	2. Never smoked	1892	1892.0
smoking status	Nominal	0%	0%	4	3. Smokes	789	789.0
smoking status	Nominal	0%	0%	4	4. Unknown	1544	1544.0

Table 3. Details of the *nominal* attributes of the stroke dataset under different settings.

wards no stroke class. This issue will significantly affect the overall classification results, so we decided to create a sample from the dataset(*Under-sampling the majority class)*. Due to the restrictions of this project guidelines, we are supposed to use a dataset with more than 1000 instances. Since we need to keep 249 yes instances, we used *spread subsample* filter under supervised filter of weka and set the *distribution Spread* to 3.94. Applying this technique, we acquired total 1230 instances and the proportion of yes/yno instances is 3.94 which is still high; however, we assume this subsample is the best option for our dataset. Note that we did not change any other property of the above mentioned filter.

We first show the summary of the original and the subsampled dataset in Table 1. Then we show the details of two different set of attributes with type of the stroke dataset in Table 2 and Table 3, respectively.

Algorithm	Correctly classified	Accuracy	F-measure for stroke class
Naive Bayes	4585	89.726%	0.215
Logistic Regression	4862	95.1468 %	0.004
Multi-layer Perceptron	4806	94.0509%	0.095
IBK(k=10)	4861	95.1272 %	Undefined
Meta AdaBoost M1	4861	95.1272 %	Undefined
OneR	4861	95.1272 %	Undefined
J48	4861	95.1272 %	Undefined

Table 4. Stroke prediction results using different machine learning algorithms on the original stroke dataset.

Algorithm	Correctly classified	Accuracy	F-measure for stroke class
Naive Bayes	967	78.6179 %	0.533
Logistic Regression	992	80.6504 %	0.425
Multi-layer Perceptron	959	77.9675 %	0.386
IBK(k=10)	987	80.2439 %	0.198
Meta AdaBoost M1	986	80.1626 %	0.378
OneR	996	80.9756 %	0.397
J48	980	79.6748 %	0.432

Table 5. Stroke prediction results using different machine learning algorithms on the sub-sampled stroke dataset.

2. Algorithms Used

In this work, we used seven different machines learning based algorithms including Naive Bayes, Logistic Regression, Multi-layer Perceptron, IBK, Meta Adaboost, OneR, and J48. Note that we used the *weka* data mining toolbox for using all the algorithms. We choose different algorithm to show the capacity of different algorithms for binary classification task. We report classification results under two different set of stroke data (original and sub-sampled) and all testing was done with 10-fold cross-validation.

The overall classification results using all the algorithms on original and sub-sampled datasets are presented in Table 4 and Table 5, respectively.

2.1. Stroke Prediction Using Naive Bayes Algorithm

We first applied a Naive Bayes classifier on the original dataset before sampling. We received 89.726% accuracy and an F-measure of 0.215 for minority class. This shows in spite of the good accuracy the classifier is not able to classify a good portion of stroke instances. Then we applied a Naive Bayes classifier on the sampled population, and we received 78.6179% accuracy but higher f-measure of 0.533. Out of 249 stroke instances the classifier was able to correctly classify 150 instances.

2.2. Stroke Prediction Using Logistic Regression

Logistic Regression algorithm achieves 95.15% classification accuracy on original dataset. However It only correctly classifies one stroke instance, which yields to a very low f-Measure of 0.008 for minority class. However, when we use the sub-sampled dataset it achieves a lower accuracy of 80.6504% and a more acceptable F-measure of 0.425 for minority class .

2.3. Stroke Prediction using Multi-layer Perceptron

Multi-layer perceptron is a small neural network which consists of an input layer, hidden layers, and output layer. Multi-layer perceptron algorithm achieves 94.0509% classification accuracy on original dataset, and F-measure of less than 0.1 ; however, when we use the sub-sampled dataset it yields a 77.9675% classification accuracy and an F-measure higher than 0.4 . These results clearly implies that similar to the Naive Bayes model and Logistic Regression, Multi-layer perceptron also performs well on classifying no stroke instances in the original dataset as the size of the dataset is biased towards the majority class. And under-sampling the dataset leads to more balanced results.

2.4. Stroke Prediction Using IBK

IBK(k=10) algorithm achieves 95.1272% classification accuracy on the original dataset and F-measure of 0.008; however, when we use the sub-sampled dataset it yields 80.2439% classification accuracy and F-measure of 0.198. These results clearly implies that similar to the other models, IBK also performs well on the original dataset for classifying the majority class, however it needs more data to make a better prediction for minority class.

2.5. Stroke Prediction Using Meta Adaboost M1

Meta Adaboost M1 algorithm achieves 95.1272% classification accuracy on original dataset and an undefined value for F-measure. This classifier is not able to correctly classify any stroke instances.Therefore, the value for F-measure and precision of the minority class is undefined ; however, when we use the sub-sampled dataset this classifier achieves a 80.1626% classification accuracy and 0.435 value for F-Measure which is very promising compared to its performance on the original dataset. In addition, It correctly classifies 94 instances of minority class which shows that sub-sampling improves the performance of Adaboost M1 for classifying stroke instances on the given dataset.

2.6. Stroke Prediction Using OneR

OneR algorithm achieves 95.1272% classification accuracy on the original dataset and an undefined F-measure value for the minority class ; however, after sampling it acquires an accuracy of 80.9756% and F-measure of above 0.3. It classifies 0 instances of stroke class on the original dataset, however on the sampled population this will increased to 77 instances of minority class.

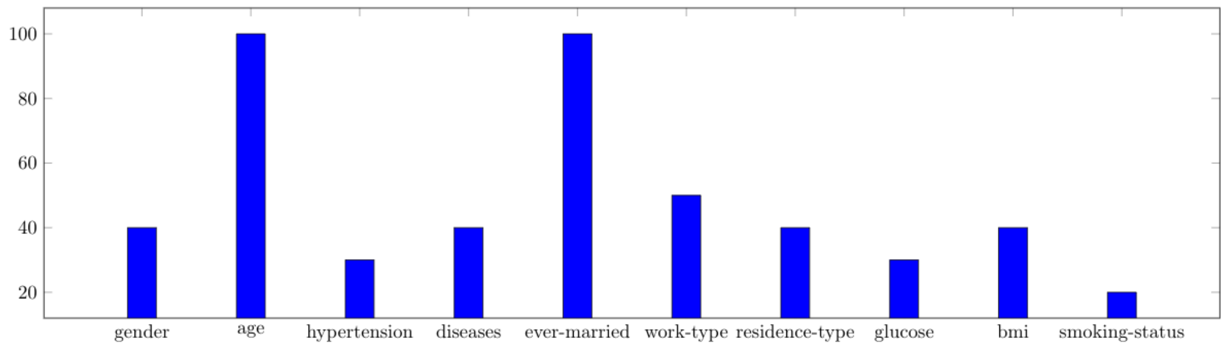

Figure 1. Illustration of scheme-specific attribute selection with *forward* direction
. X axis shows attribute names , Y axis shows the number of times that the attribute appears in 10 Folds (in percentage)

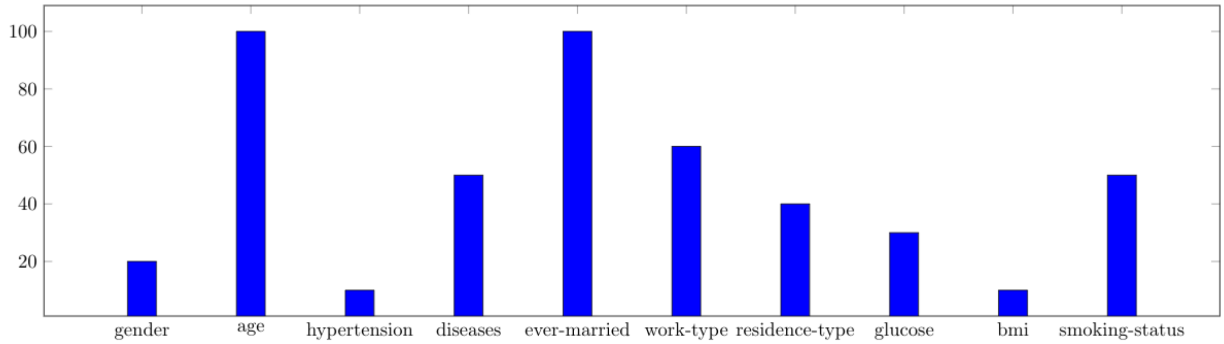

Figure 2. Illustration of scheme-specific attribute selection with *backward* direction. X axis shows attribute names , Y axis shows the number of times that the attribute appears in 10 Folds (in percentage)

2.7. Stroke Prediction Using J48

J48 algorithm achieves 95.1272% classification accuracy on original dataset and an undefined F-measure value ; however, when we use the sub-sampled dataset yields 79.6748% classification accuracy and an F-measure of 0.432 for minority class. The classifier is capable of correctly classifying 95 instances of stroke class which is a great improvement compared to its results on the original dataset.

3. Attribute Selection

In the previous sections, we have reported classification results on two different versions of the stroke dataset. We used all the given attributes of the datasets for classification task. Now we perform the similar set of experiments but using selected attributes under two different settings (scheme-specific and scheme-independent). We choose two different schemes to select attributes as adding an irrelevant attribute can significantly degrade the classification performance for some algorithms. Note that we perform attribute selection experiments only on the sub-sampled data.

Algorithm	Correctly classified	Accuracy	F-measure for stroke class
Naive Bayes	951	77.3171 %	0.490
Logistic Regression	1005	81.7073 %	0.404
Multi-layer Perceptron	1004	81.626 %	0.405
IBK(k=10)	1003	81.5447 %	0.371
Meta AdaBoost M1	986	80.1626 %	0.522
OneR	996	80.9756 %	0.397
J48	1009	82.0325 %	0.429

Table 6. Stroke prediction results using different machine learning algorithms on the sub-sampled stroke dataset under scheme-specific selected attributes (Age and ever married) setting.

3.0.1 Scheme-specific Attribute selection

We first report experimental results for classification task under scheme-specific attribute selection setting. In this scheme, attribute selection is implemented as wrapper

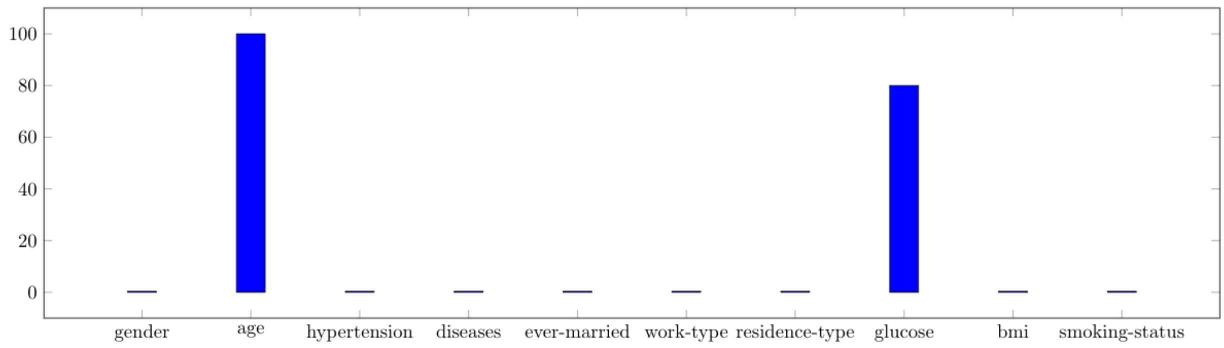

Figure 3. Illustration of scheme-independent attribute selection. X axis shows attribute names , Y axis shows the number of the time that the attribute appears in 10 Folds (in percentage).

around learning scheme and essential for learning decision tables. However, it is time consuming, adds factor k^2 even for greedy approaches with k attributes.

For this part we chose *WrapperSubsetEval* for attribute evaluator and used J48 as classifier. We used the default number of folds and set the threshold to -1 . We also used BestFirst as the searching method and tried both Forward and backward direction. We tried Cross-validation with 10 Folds as the attribute selection mode and we received the following results which are shown in Fig. 1 and Fig. 2, respectively.

The overall classification results using all the algorithms on sub-sampled dataset under scheme-specific attribute selection are presented in Table 6. Note that in this setting we choose *age* and *ever-married* attributes only for the stroke classification task. Interestingly, using only using two attributes can outperforms the similar algorithm which is trained on the dataset with all the attributes. The results are consistent for most of the algorithms. For instance, the Logistic regression algorithm achieves 81.7% accuracy and F1 measure of 0.4 using scheme-specific attribute selection while it yields 80.6% accuracy with same F1 measure using all the attributes.The greatest improvement is seen in case of J48 classifier with 82.0325% accuracy and F1 measure of 0.429. In addition, these results clearly support the hypothesis that not all the attributes play an important role in the decision making process for the classification Task.

3.0.2 Scheme-independent Attribute selection

Now we extend the previous experiments by selecting attributes based on scheme-independent attribute selection technique. This techniques selects attributes based on the filtering approaching where assessment is based on general characteristics of the data. It first finds subset of attributes that is enough to separate all the instances.

Algorithm	Correctly classified	Accuracy	F-measure for stroke class
Naive Bayes	978	79.5122 %	0.397
Logistic Regression	1011	82.1951 %	0.475
Multi-layerPerceptron	1005	81.7073 %	0.508
IBK	992	80.6504%	0.411
Meta AdaBoost M1	982	79.8374 %	0.415
OneR	996	80.9756 %	0.397
J48	990	80.4878 %	0.417

Table 7. Stroke prediction results using different machine learning algorithms on the sub-sampled stroke dataset under scheme-independent selected attributes setting i.e age and avg glucose level.

For scheme-independent attribute selection, we chose *CfsSubsetEval* as the evaluator and used search method BestFirst. We tried both forward and backward direction with cross-validation 10 folds (default setting). The results are shown in Fig. 3.

The overall classification results using all the algorithms on sub-sampled dataset under scheme-sindependent attribute selection are presented in Table 7. Note that in this setting we choose *age* and *avg glucose level* attributes only for the stroke classification task. Interestingly, only using these two attributes can outperforms the similar algorithm which is trained on the datasets with all the attributes. The results are consistent for most of the algorithms. For instance, the Logistic regression algorithm achieves 82.2% accuracy and F1 measure of 0.475 using scheme-independent attribute selection while it yields 80.6% accuracy with same F1 measure using all the attributes. In addition, these results clearly support the hypothesis that not all the attributes play an important role in the decision making process for the classification task. Both schemes identify age as a very important attribute, however scheme-specifics identify ever-married and scheme-

Algorithm	Correctly classified	Accuracy
Naive Bayes	964	78.374 %
Logistic Regression	995	81.1382 %
Multi-layer Perceptron	942	76.5854 %
IBK	963	78.2927%
OneR	999	81.2195%
J48	998	81.1382 %

Table 8. Stroke prediction results using different machine learning algorithms on the sub-sampled stroke dataset using unsupervised discretization.

independent identifies avg-glucose as important attributes.

Compared to the scheme-specific attribute selection technique, scheme-independent attribute selection selects attributes which plays more important role in classification task , resulting in overall better accuracy and F-measure.

4. Performance comparison between all attributes vs. selected attributes.

We now compare the overall results obtained with all the attributes vs. selected attributes. Logistic regression achieves the highest performance in terms of accuracy and F measure on all attributes. After attribute selection on sampled data the highest accuracy is achieved by j48 for scheme specific with 82.04% accuracy and F1 measure of above 0.4. In addition logistic regression with 81.7073% accuracy and F1 measure of 0.508 has the best performance after scheme independent attribute selection. Interestingly, we observed that a proper selection of attributes can improve the performance of J48 and reduce the size of the tree from 132 to 9.

5. Discretization

We apply two different techniques under discretization (supervised and unsupervised) for further analysis.

5.1. Unsupervised Discretization

We tried both equal-interval binning by using Discretize filter in Weka (useEqualFrequency set to False) and equal frequency discretization by using the same filter (useEqualFrequency set to True). In addition ,we repeated our experiment with different numbers of bins. The results of this analysis are presented in Table 8. Interestingly, setting the number of bins to 11 achieves the best results. However, after experimenting we realized that Discretization alone will not change the result very dramatically. Therefore, we applied discretization on the selected attributes to see if any improvement can be achieved. We applied discretization on attributes age and ever-married and observed a slight improvement in F1 measure in case of Logistic Regression. We repeat the same experiment with equal frequency discretization technique and we did not observe any improvement. Note that in this experiment we had to apply some modification on the J48 pruning setting to be able to have a proper tree structure. For this purpose we set a 0.5 confidence factor and set the max count to 5.

We also present the confusion matrix for all the algorithms based on classification performance.

Prediction outcome: Naive Bayes

		No stroke	**Stroke**	**total**
actual value	**No stroke**	832	149	No stroke′
	Stroke	117	132	Stroke′
	total	No stroke	Stroke	

Prediction outcome: J48

		No stroke	**Stroke**	**total**
actual value	**No stroke**	908	73	No stroke′
	Stroke	159	90	Stroke′
	total	No stroke	Stroke	

Prediction outcome: Logistic Regression

		No stroke	**Stroke**	**total**
actual value	**No stroke**	905	76	No stroke′
	Stroke	159	90	Stroke′
	total	No stroke	Stroke	

Prediction outcome: Multi-layer Perceptron

		No stroke	Stroke	total
actual value	No stroke	849	132	No stroke′
	Stroke	156	93	Stroke′
	total	No stroke	Stroke	

Prediction outcome: OneR

		No stroke	Stroke	total
actual value	No stroke	897	84	No stroke′
	Stroke	147	102	Stroke′
	total	No stroke	Stroke	

Prediction outcome: IBK

		No stroke	Stroke	total
actual value	No stroke	906	75	No stroke′
	Stroke	192	57	Stroke′
	total	No stroke	Stroke	

5.2. Supervised Discretization

We further applied supervised discretization [2] techniques[2] for this classification task. Note that supervised discretization technique takes the class into consideration when discretizing. In order to do this we used a Filtered Classifier under the meta classifier of Weka. We set J48 as the classifier and supervised Discretize as the filter. We observed an accuracy of 80.6504% and F-Measure of 0.402.

Supervised Discretization Prediction outcome: J48

		No stroke	Stroke	total
actual value	No stroke	956	25	No stroke′
	Stroke	205	44	Stroke′
	total	No stroke	Stroke	

6. Evaluation Metrics

We used the overall accuracy and F-measure to report the classification performance. As mentioned earlier in this paper, correctly classifying instances of minority class is more important than that of majority class due to the nature of the problem. For instance, It is more crucial to correctly predict if a person is at risk of having a stroke. Detecting a person at risk of stroke as someone who is not at risk of stroke might have some serious consequences. Therefore, we are more interested in reducing the number of incorrectly classified stroke instances or reducing the False Negative rate. Although, Recall can be another good measurement when False Negative error is more costly, we chose to report f-measure since it represents a balance between recall and precision measures. Therefore, achieving a higher value of F- measure is more valuable for us than a high accuracy value. The F-measure can be calculated using formula(1).

F-measure= 2* Precision* Recall/precision+Recall (1)

Sampled Prediction outcome: oneR

		No stroke	Stroke	total
actual value	No stroke	919	62	No stroke′
	Stroke	172	77	Stroke′
	total	No stroke	Stroke	

The above confusion matrix determines the accuracy of oneR model. In the confusion matrix, No stroke refers to class 0 while stroke refers to class 1. Rows denotes expected class values and columns refers to predicted class values. For instance, the oneR model thinks 919 instances are correctly predicted, so the true positive rate is 919. 62 instances are predicted as b but the true label is a. Similarly, the oneR model assumes 172 instances as a but the true label is b which is refereed to as false positive (FP). 77 instances are predicted as b by the model but the true labels are b which is considered as true negative (TN). We can calculate the overall error rate and the success rate based on the

[2] More Data Mining with Weka (2.2: Supervised discretization and the FilteredClassifier). Retrieved from https://www.youtube.com/watch?v=DBkdvJQDJ5c

metrics.

$$\begin{aligned} TP_{rate} &= \frac{TP}{TP+FN} = \frac{919}{919+62} = 0.93 \\ FP_{rate} &= \frac{FP}{FP+TN} = \frac{172}{172+77} = 0.69 \end{aligned} \tag{1}$$

From the above calculation we can see that TP rate is significantly higher than FP rate which indicates that the performance of the oneR model is better.

scheme independent: Age and glucose(Multilayer)

		No stroke	**Stroke**	**total**
actual value	**No stroke**	889	92	No stroke′
	Stroke	133	116	Stroke′
	total	No stroke	Stroke	

The above confusion matrix determines the accuracy of the model.For instance, the model thinks 889 instances are correctly predicted, so the true positive rate is 889. 92 instances are predicted as b but the true label is a. Similarly, the model assumes 133 instances as a but the true label is b which is refereed to as false positive (FP). 116 instances are predicted as b by the model but the true labels are b which is considered as true negative (TN). We can calculate the overall error rate and the success rate based on the metrics.

$$\begin{aligned} TP_{rate} &= \frac{TP}{TP+FN} = \frac{889}{889+92} = 0.91 \\ FP_{rate} &= \frac{FP}{FP+TN} = \frac{133}{133+116} = 0.55 \end{aligned} \tag{2}$$

From the above calculation we can see that TP rate is significantly higher than FP rate which indicates that the performance of the Multilayer model with attribute selection is good but it is not better than OneR model on (sub sampled) data.

scheme specific: Age and ever-married(J48)

		No stroke	**Stroke**	**total**
actual value	**No stroke**	926	55	No stroke′
	Stroke	166	83	Stroke′
	total	No stroke	Stroke	

The above confusion matrix determines the accuracy of the J48 model.For instance, the model thinks 926 instances are correctly predicted, so the true positive rate is 926. 55 instances are predicted as b but the true label is a. Similarly, the model assumes 166 instances as a but the true label is b which is refereed to as false positive (FP). 83 instances are predicted as b by the model but the true labels are b which is considered as true negative (TN). We can calculate the overall error rate and the success rate based on the metrics.

$$\begin{aligned} TP_{rate} &= \frac{TP}{TP+FN} = \frac{926}{926+55} = 0.94 \\ FP_{rate} &= \frac{FP}{FP+TN} = \frac{166}{166+83} = 0.67 \end{aligned} \tag{3}$$

From the above calculation we can see that TP rate is significantly higher than FP rate which indicates that the performance of the J48 model with attribute selection(age and ever-married) is better than both scheme independent and sub sampled data with all attributes.

7. Discussion

7.1. Using Baseline classifier

One way to see which model gives better results in terms of accuracy is to compare their accuracy with the baseline. In order to find the baseline we can use the ZeroR model. From the course lectures we know that ZeroR finds the majority class and predicts every new instance based on that, so if we use the training set as a test set we will be able to see the baseline value which is 79.7561% (981/1230). If we compare other models' accuracy (using cross validation) with baseline before applying any attribute selection or discretization , we can see that NB with accuracy of 78.6179% performs slightly worse than baseline. Logistics Regression with accuracy of 80.6504% performs better than the baseline. KNN with k=10 has accuracy of 80.2439% which is better than baseline. OneR classifier gives us 80.9756% which is higher than baseline. And J48 gives us 79.6748% (tree size:132) which is lower than the baseline.

8. Conclusion

In this paper we applied different data mining techniques on a stroke classification dataset. We applied a number of transformation techniques on the original dataset .The dataset was highly biased towards no stroke class. In order to tackle this problem we undersample the majority class. We applied 7 different classifiers on the original dataset and the sampled population and compared their F measure and accuracy. We also applied two different attribute selection techniques and applied all 7 classifiers on the sampled dataset with selected attributes. Finally, we tried supervised and unsupervised discretization on the sampled dataset with a subset of attributes. The original dataset was very biased towards no stroke class, however we showed that undersampling can alleviate this issue to some extent. In general Logistic regression archives the best result on the subsampled dataset, However J48 has the best performance with attributes selected from Scheme-specific Attribute selection.

On the other hand, Logistic regression shows the best performance on the attribute selected by Scheme-independent Attribute selection. In addition we observed that discretization does not change the previous results very significantly. We also observed that Meta Adaboost M1 gives the best F-measure for minority class when selecting age and ever married attributes.

9. Workload Distribution

Work	Saghar	Fabliha
Report write up	✓	✗
Latex write up	✗	✓
Discussion about the results	✓	✓
Prepossessing	✓	✗
Attribute Selection(each of us worked on one method)	✓	✓
Logistic Regression	✓	✗
J48	✓	✗
IBK	✓	✗
Discretization	✓	✗
F-measure	✓	✗
Evaluation metrics	✓	✓
NB Classifier	✗	✓
Meta AdaBoost M1 Classifier	✗	✓
Perceptron Classifier	✗	✓
OneR Classifier	✗	✓
Performance Comparison	✗	✓
Abstract	✗	✓

Table 9. Summary of the workload distribution in terms of different workload features.

References

[1] Fedesoriano. Stroke prediction dataset. https://www.kaggle.com/fedesoriano/stroke-prediction-dataset, 2021. 1

[2] Ian Witten. More data mining with weka (2.2: Supervised discretization and the filteredclassifier). https://www.youtube.com/watch?v=DBkdvJQDJ5c, 2014. 7