PROJECT REPORT: CS 508

Experimenting data mining tasks on an arbitrary dataset

Written by:

Omid Jafari

Sainikhitha Thooti

Michael Garcia

Michael Lyon

Department of Computer Science

New Mexico State University

December 4 2018

Abstract: Autism spectrum disorder (ASD) is a neurodevelopmental condition that is characterized by challenges with social skills, repetitive behaviors, speech, and nonverbal communication. Healthcare costs are very high for an individual with ASD, but an early diagnosis can significantly reduce these costs. Waiting times for ASD are long, and the procedures needed are also very expensive. As the number of ASD cases each year increases around the world, it has become clear that we need easier, more effective screening methods. Datasets related to ASD are rare, and the ones that are around mostly focus on genetics. The dataset we found on ASD screening of adults by Fadi Fayez Thabtah from UCI Machine Learning repository contains 21 attributes and 704 instances that can be used for further analysis. We believe we can use this dataset to help determine whether an individual has ASD or not. The objective of this project is to find a classifier which can predict an unknown instance with an acceptable accuracy and to cluster the data into several groups, so that they could have meaning. Moreover, we defined visualizing data as an objective since it can help us better understand the dataset and data mining results.

Table of contents

1. Introduction

We started off by cleaning the dataset and fixing issues such as missing values, outliers, and false predictors. Afterwards, since numeric attributes cannot help us in the predictions, we had to discretize all of them into several bins. Furthermore, as some attributes may have a weaker correlation to the class attribute, it seemed reasonable to remove them and save training time. The running time on the dataset we had was inconsiderable as the size of it was small. However, for the large datasets, attribute selection is an important step to save time and have a simpler model on the data.

After doing attribute selection, we were left with two more tasks: classification and clustering. The appropriate way of running these tasks is to first split the dataset into train, test, and validation sets. However, since our dataset did not have enough instances to perform these operations, we used the k-fold cross validation method. Finally, we visualized the dataset using different methods such as scatterplots, and parallel coordinates.

1.1 Overview Of Dataset

For our analysis, we used the Autism Screening Adult from UCI repository. This dataset has 704 instances, 21 integer attributes, and contains missing values. *Table(1)* contains a description of each attribute along with their types.

Table(1) Attribute Information

Attribute	**Type**	**Description**
A1_Score(Attribute 1)	Nominal	The answer code of the question based on the screening method used
A2_Score(Attribute 2)	Nominal	The answer code of the question based on the screening method used
A3_Score(Attribute 3)	Nominal	The answer code of the question based on the screening method used
A4_Score(Attribute 4)	Nominal	The answer code of the question based on the screening method used
A5_Score(Attribute 5)	Nominal	The answer code of the question based on the screening method used
A6_Score(Attribute 6)	Nominal	The answer code of the question based on the screening method used
A7_Score(Attribute 7)	Nominal	The answer code of the question based on the screening method used
A8_Score(Attribute 8)	Nominal	The answer code of the question based on the screening method used
A9_Score(Attribute 9)	Nominal	The answer code of the question based on the screening method used
A10_Score(Attribute 10)	Nominal	The answer code of the question based on the screening method used

Age(Attribute 11)	Numeric	Age in years
Gender(Attribute 12)	Nominal	Male or Female
Ethnicity(Attribute 13)	Nominal	List of common ethnicities in text format
Jaundice(Attribute 14)	Nominal	Whether the case was born with jaundice
Family member with autism(Attribute 15)	Nominal	Whether any immediate family member has a autism
Country of residence (Attribute 16)	Nominal	List of countries in text format
Used the screening app before(Attribute 17)	Nominal	Whether the user has used a screening app
Age Description (Attribute 18)	Nominal	Whether the age is 18 or more
Result(Attribute 19)	Numeric	The final score obtained based on the scoring algorithm of the screening method used. This was computed in an automated manner
Relation(Attribute 20)	Nominal	Parent, self, caregiver, medical staff, clinician ,etc.
Class/ASD(Attribute 21)	Nominal	NO,YES

2. Preprocessing

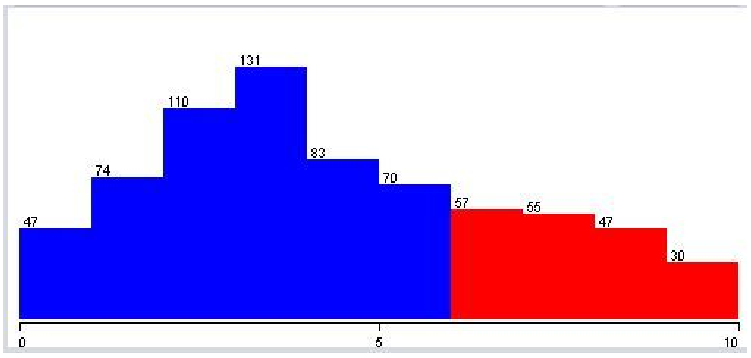
Data goes through a series of steps during preprocessing:

1. Data Cleaning: Data is cleansed through processes such as filling in missing values, smoothing the noisy data, or resolving the inconsistencies in the data.

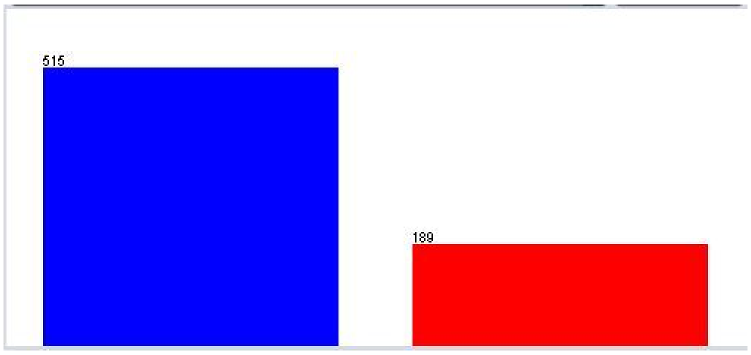
2. Data Discretization: Involves the reduction of a number of values of a continuous attribute by dividing the range of attribute intervals.

2.1 False Predictors

By looking at the Weka visualization of each attribute, we realized that there were some unneeded attributes in the dataset. In other words, there existed an attribute ("result") which had 100% correlation with the class. Knowing this information, we could conclude that it was a duplicate class. Therefore, we decided to remove this attribute from the dataset. The visualization of this attribute and the class can be seen in *figure(1a)*. We used the "Remove" filter from the unsupervised, attribute section of Weka to remove the mentioned attribute.

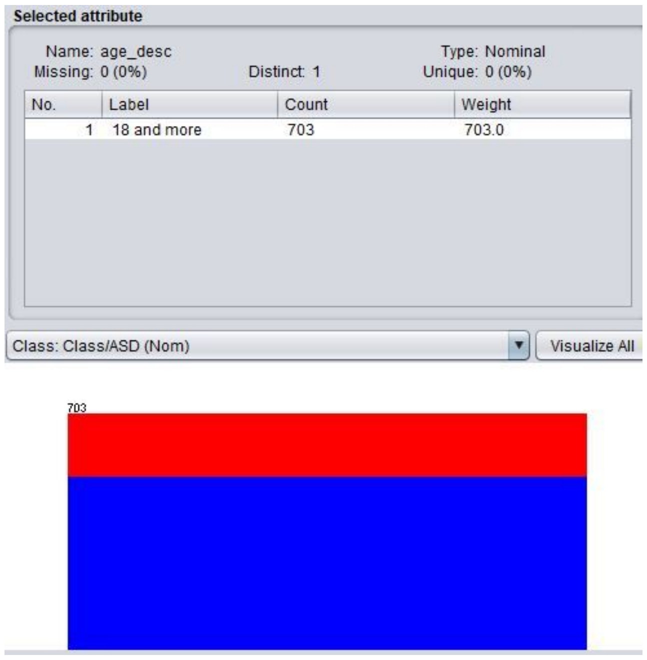


Figure(1a) duplicate attribute visualization



Figure(1b) class visualization

Furthermore, there was another attribute called "age_desc" which as shown in *figure(1c)*, only had one value for all instances. Therefore, we also removed this attribute from the dataset.



Selected attribute

Name: age_desc Type: Nominal
Missing: 0 (0%) Distinct: 1 Unique: 0 (0%)

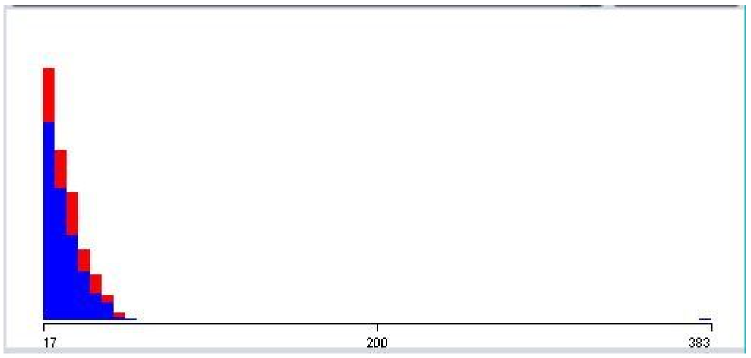
No.	Label	Count	Weight
1	18 and more	703	703.0

Class: Class/ASD (Nom)

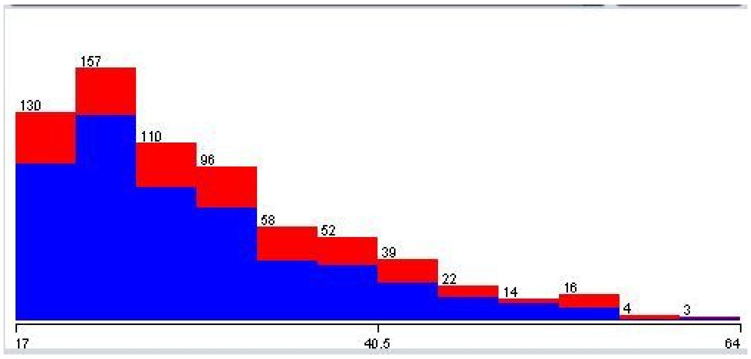
Figure(1c) attribute "age_desc" visualization

2.2 Outliers

Using the same method for detecting false predictors, and as it can be seen in *figure(2a),* we found there was an outlier in the "age" attribute. The value of this outlier was 383. To remove this instance from the dataset, we used the unsupervised, instance-related section and chose “RemoveWithValues” filter. Using this filter, we can define a split point (number), and anything less than that number will be kept in the dataset. We used the split value of 100. *Figure(2b)* shows the resulting filtered dataset.



Figure(2a) age attribute outlier

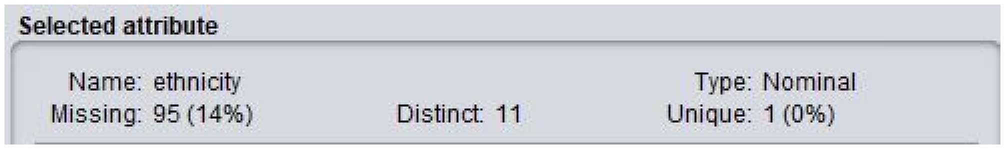


Figure(2b) filtered age attribute

2.3 Missing Values

When we originally picked this dataset, we knew it had missing values. To find which values were missing, we checked the “Missing” field in “Selected Attribute” section for all attributes.

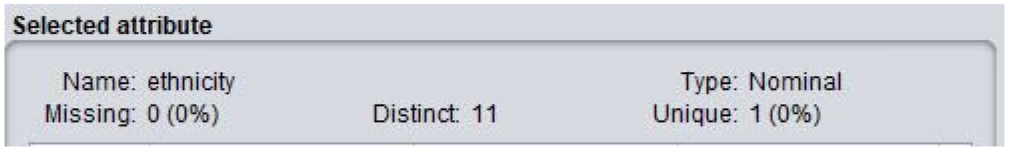
We found out that the attribute “age” had 2 missing values, “ethnicity” had 95 missing values, and “relation” had 95 missing values. There are several techniques to solve the issue of replacing missing attributes. We decided to replace each missing value with the mean values of its corresponding attribute. To do so, we used the “ReplaceMissingValues” filter from unsupervised, attribute-related filters. This filter automatically replaces all missing values with the mean value of its corresponding attribute. *Figure(3a)* and *figure(3b)* show the “ethnicity” attribute before and after filtering.



Selected attribute

Name: ethnicity
Missing: 95 (14%)
Distinct: 11
Type: Nominal
Unique: 1 (0%)

Figure(3a) unfiltered ethnicity attribute



Selected attribute

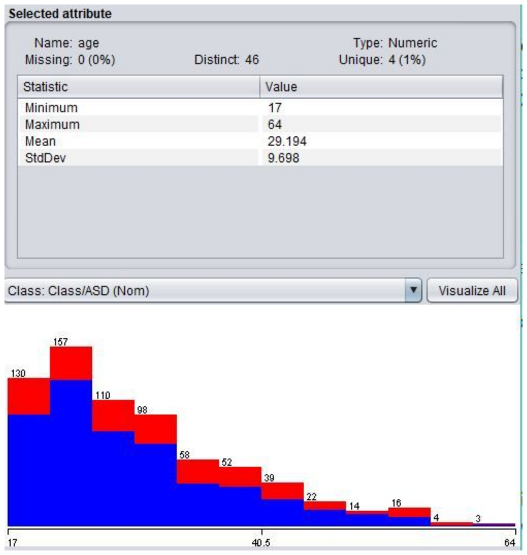
Name: ethnicity
Missing: 0 (0%)
Distinct: 11
Type: Nominal
Unique: 1 (0%)

Figure(3b) filtered ethnicity attribute

2.4 Discretization

The only numeric attribute we were left with was “age”, and the task was to discretize this attribute into several bins. Discretization allows us to use more classifiers and have a better prediction model. To discretize this attribute, we used the “Discretize” filter from unsupervised, attribute-related filters.

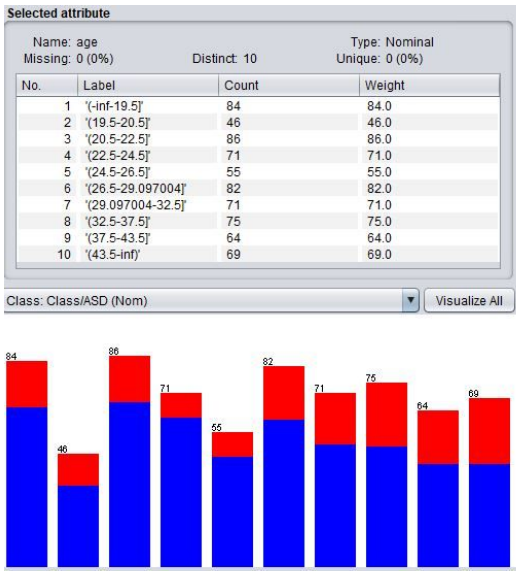
Figure(4a) shows that most of the dataset was consisted of younger people. We decided to use the equal-height discretization strategy to get more balanced bins with the default number of bins (10). *Figure(4b)* shows the result of discretization.


Selected attribute

Name: age
Missing: 0 (0%)
Distinct: 46
Type: Numeric
Unique: 4 (1%)

Statistic	Value
Minimum	17
Maximum	64
Mean	29.194
StdDev	9.698

Figure(4a) “Age” attribute graph


Selected attribute

Name: age
Missing: 0 (0%)
Distinct: 10
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	'(-inf-19.5]'	84	84.0
2	'(19.5-20.5]'	46	46.0
3	'(20.5-22.5]'	86	86.0
4	'(22.5-24.5]'	71	71.0
5	'(24.5-26.5]'	55	55.0
6	'(26.5-29.097004]'	82	82.0
7	'(29.097004-32.5]'	71	71.0
8	'(32.5-37.5]'	75	75.0
9	'(37.5-43.5]'	64	64.0
10	'(43.5-inf)'	69	69.0

Figure(4b) Filtered “Age” attribute with equal-height discretization

3. Attribute Selection

As mentioned before, attribute selection is an important step in data mining, especially for larger data sets. Since Weka has several attribute selection algorithms, we tried several of them and listed the selected attributes of each one as seen in *table(2)*. We then chose the common attributes as our final attributes.

Table(2) Attribute selection results

Attribute Evaluator	**Search Method**	**Selected attributes**
ClassifierAttributeEval	Ranker	19,5,7,6,4,9,3,2,8,10,15,17,16,14,11,13,12,1
CorrelationAttributeEval	Ranker	9,6,5,4,3,10,7,2,1,8,15,13,16,14,12,11,17,19
GainRatioAtrributeEval	Ranker	9,6,5,4,3,10,1,7,2,16,8,15,13,14,17,11,12,19
InfoGainAttributeEval	Ranker	9,6,5,16,4,3,10,7,1,2,13,8,15,11,14,12,17,19
OneRAttributeEval	Ranker	9,6,16,19,7,3,4,2,8,10,11,15,14,1,12,5,17,13
CfsSubsetEval	GreedyStepWise	1,2,3,4,5,6,7,8,9,10,16
CfsSubsetEval	BestFirst	1,2,3,4,5,6,7,8,9,10,16

A1_Score(attribute 1) ,

A2_Score(attribute 2),

A3_Score(attribute 3),

A4_Score(attribute 4),

A5_Score(attribute 5),

A6_Score(attribute 6),

A7_Score(attribute 7),

A8_Score(attribute 8),

A9_Score(attribute 9),

A10_Score(attribute 10),

age(attribute 11),

gender(attribute 12),

ethnicity(attribute 13),

jaundice(attribute 14),

family member with autism(attribute 15),

country-of-residence(attribute 16),

used-app-before(attribute 17),

age_desc(attribute 18),

relation(attribute 19),

Out of these 19 attributes,these (1,2,3,4,5,6,7,8,9,10,16) ,11 attributes that are common between all of the algorithms. Using this information, we chose attributes 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 16 as our selected attributes, removing all the other attributes. The final dataset has 12 attributes: 11 from the attribute selection step, and one class attribute.

4. Classification

There are many classification methods and algorithms in Weka for different types of attributes and datasets. We tried all classifiers that were suitable for our chosen dataset and compared them using the accuracy metric. For each classifier, we used the default parameters, and since our dataset was small we used the k-fold cross validation method with k=10. To perform the task, we used the experimenter tab from Weka. From there, all we had to do was to load the preprocessed dataset and add the classifiers and hit the run button.

We used the following classifiers: *Naive Bayes, Logistic, SGD, SMO, IBk, OneR, ZeroR, J48, Id3, and RandomForest.* For a better comparison, besides having the results in the table format, we also used charts. Table (3) and Figure (5) show the results.

Table(3) Classification Comparison Results with Default Parameters

Classifiers	**Accuracy**
NaiveBayes	96.74%
Logistic	97.95%
SMO	100%
SGD	99.64%
IBK	96.36%
OneR	83.28%
ZeroR	73.11%
J48	91.58%
Id3	81.39%
RandomForest	92.16%

Table(4) Classification Comparison Results with Different Parameters

Classifiers	Parameters	Accuracy
NaiveBayes	donotcheckCapabilities = true	96.74% (+0.0%)
	useSupervisedDiscretization = true	96.74% (+0.0%)
Logistic	donotcheckCapabilities = true	97.95% (+0.0%)
	useConjugateGradientDescentTo = True	98.02% (+0.07%)
	maxlts = -273.11	73.11% (-24.82%)
SGD	donotcheckCapabilities = true	99.64% (+0.0%)
	lossfunction = log Loss	99.30% (-0.34%)
	epochs = 1000	99.67% (+0.03%)
SMO	donotcheckCapabilities = true	100% (+0.0%)
	randomSeed = 2	100% (+0.0%)
	batchSize= 200	100% (+0.0%)
IBK	nearestNeighbourSearchAlgorithm = BallTree	96.36% (+0.0%)
	KNN = 2 (initially 1)	97.34% (+0.98%)
	batchSize = 500 (initially 100)	96.36% (+0.0%)
	donotcheckCapabilities = true	96.36% (+0.0%)
OneR	donotcheckCapabilities = true	83.28% (+0.0%)
	minBucketSize = 10 (initially 6)	83.28% (+0.0%)
	batchSize = 200 (initially 100)	83.28% (+0.0%)
ZeroR	donotcheckCapabilities = True	73.11% (+0.0%)

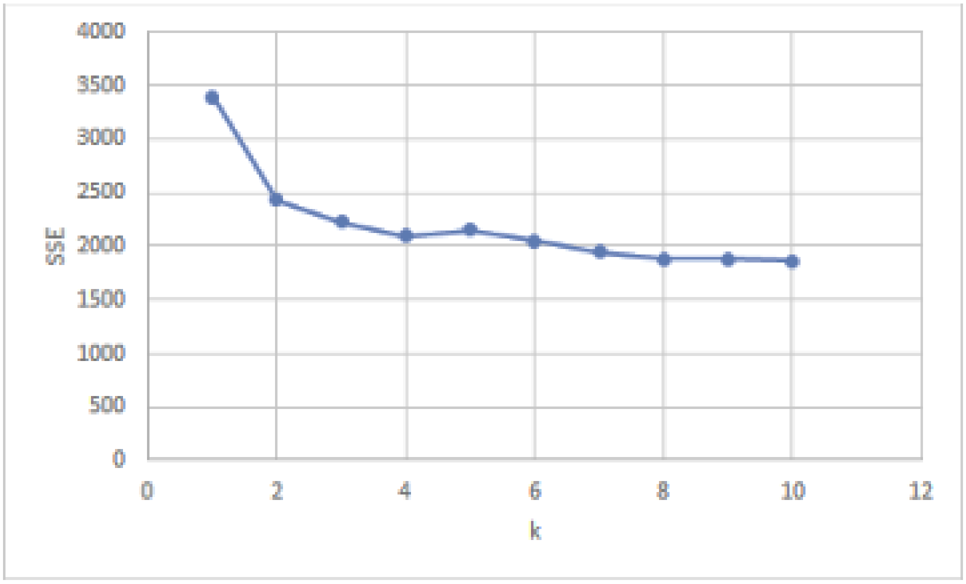
	batchSize = 200 (initially 100)	73.11% (+0.0%)
J48	confidenceFactor = 0.50 (initially 0.25)	91.88% (+0.30%)
	donotcheckCapabilities = True	91.58% (+0.0%)
Id3	donotcheckCapabilities = True	81.39% (+0.0%)
	batchSize = 200 (initially 100)	81.39% (+0.0%)
RandomForest	donotcheckCapabilities = True	92.16% (+0.0%)
	numberOfIterations = 200 (initially 100)	92.19% (+0.0%)
	seed = 2 (initially 1)	93.24% (+1.05%)

5. Clustering

There are several clustering methods such as partition-based and hierarchical clustering. Weka has several algorithms for each of these methods. To cluster our dataset, we used some of these algorithms and used accuracy (correctly clustered instances percentage) as the evaluation metric.

It is worth mentioning that clustering is an unsupervised task, and to perform this task, the class attribute should be included in the task. As a result, we used the “Classes to clusters evaluation” option to set the class attribute to be used only for evaluation. As for the parameters, default values were used. However, for partition-based methods, we have to know the number of clusters beforehand. Therefore, we first used the kMeans algorithm (with initialization method set to kMeans++) and the elbow method to find the optimal number of clusters.

To use the elbow method, all we had to do was to change the number of clusters and get the SSE value for each one. Next, we plotted the results. *Figure(6)* shows the result of elbow method. As it can be seen in the figure, the sharp change is happening at k=2 and that was the optimal number of cluster to use.

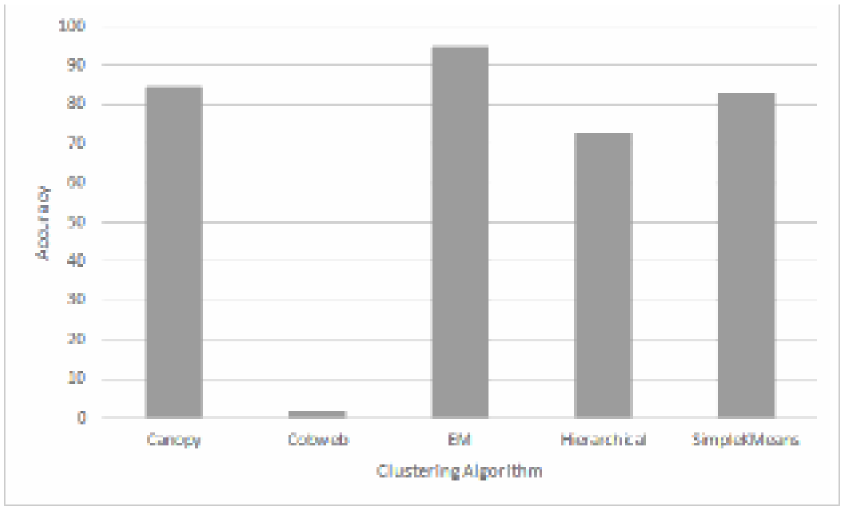


Figure(6) Elbow Method

Same as the classification task, we represent the clustering results both in table and chart formats. For the clustering task, we used *Canopy, Cobweb, EM, HierarchicalClusterer, and SimpleKMeans* algorithms. The results can be seen in *table(5)* and *figure(7)*.

Table(5) Clustering Comparison Results

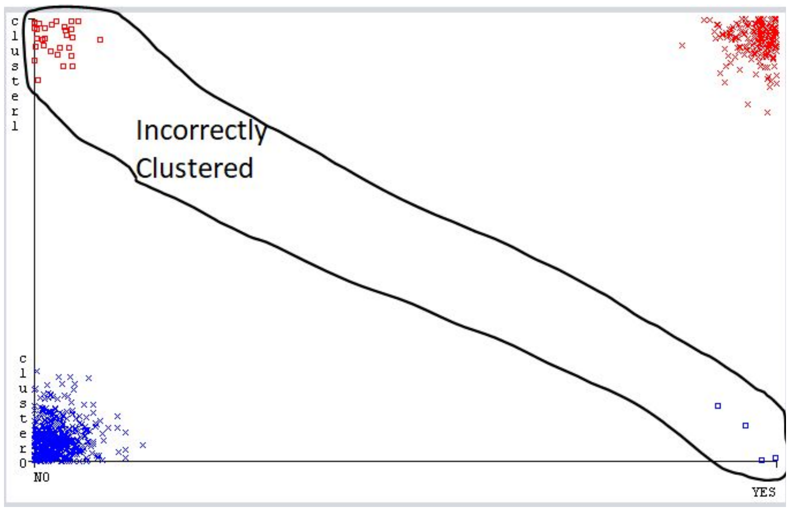
Clusterer	Accuracy
Canopy	84.495
Cobweb	1.8492
EM	95.0213
HierarchicalClusterer	72.973
SimpleKMeans	83.2148



Figure(7) Clustering Comparison Chart

From the above results, it can be inferred that *EM* and *SimpleKMeans* methods have the best accuracy. The reason is that these methods are similar to each other, with only one difference. *EM* uses membership probabilities instead of distances.

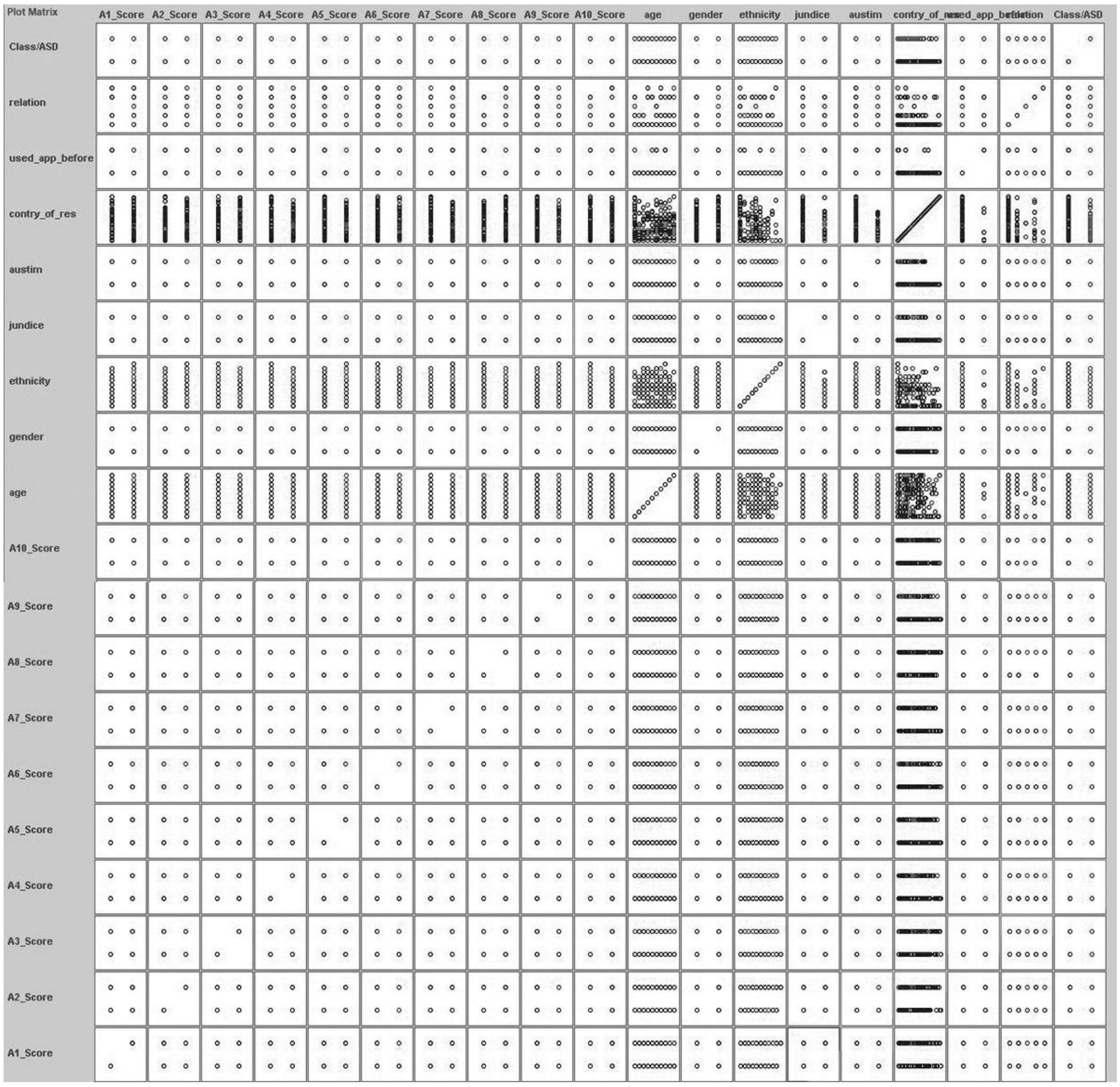
We can also visualize the cluster assignments by right clicking on specific algorithm in the Result list and choose *Visualize cluster assignment*. As you can see in *figure(8)*, we chose the actual class as the X axis and the clusters as Y axis. Anything on the y=x area is the correctly clustered instance, and the other instances are clustered incorrectly (defined by the line).



Figure(8) Clusters assignment visualization of KMeans

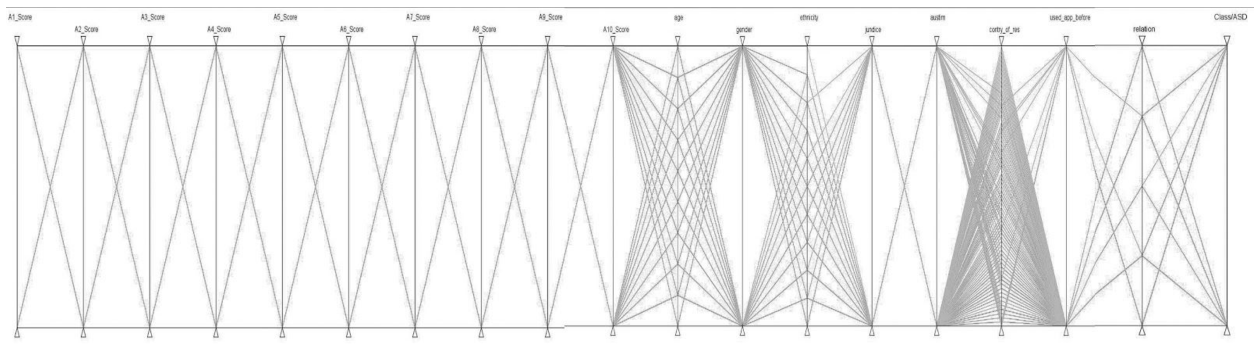
6. Visualization

Visualizing the data using scatter plots is a simple task. The scatter plots can be seen in the *Visualize* tab of Weka. *figure(9)* shows the scatter plots for our dataset.



Figure(9) Scatter Plots

To use the parallel coordinates method, we first had to install the *IPCP* package on Weka from the Package Manager. Then, we can simply open the dataset and use the *Interactive Parallel Coordinates Plot* tab to see the plot. Result can be seen in *figure(10)*.



Figure(10) Parallel Coordinates

7. Conclusion

In this project, we were tasked to do several data mining tasks on a real dataset. We found an interesting dataset which can help scientists diagnose autism in its early stages based on some gathered information from the people. However, since this dataset contained missing values and outliers, we had to first preprocess and clean the dataset.

After doing so, we did the attribute selection task in order to choose the most important attributes for the next tasks. Next, we ran the classification and clustering tasks by utilizing several algorithms available in Weka. Furthermore, we changed the parameters of each classifier and compared the accuracy.

We concluded that for the classification, SMO gives us the best accuracy, and for the clustering task, EM has the best accuracy.

For the future work, we suggest to work on a larger dataset and use separate training, validation, testing sets to correctly evaluate the classifiers and clusterers.

8. References

1. Tabtah, F. (2017). Autism Spectrum Disorder Screening: Machine Learning Adaptation and DSM-5 Fulfillment. Proceedings of the 1st International Conference on Medical and Health Informatics 2017, pp.1-6. Taichung City, Taiwan, ACM.

2. Thabtah, F.(2017). ASDTests. A mobile app for ASD screening. www.asdtests.com [accessed December 20th, 2017].

3. Thabtah, F. (2017). Machine Learning in Autistic Spectrum Disorder Behavioural Research: A Review. To Appear in Informatics for Health and Social Care Journal. December, 2017 (in press)