Data Mining Analysis Report

CP3403 – Data mining

Udaya Bhaskar Reddy Malkannagari

Student ID: 13368171

Lecturer: Mr. Kwang Lim

Contents

Abstract:

The purpose of this report is to use different methods of preprocessing and data mining techniques to gain an understanding of congressional voting patterns in the United states on various set policies and agendas. The aim is to understand the accuracy and the timeliness of the information in the chosen dataset, which is extremely important in future implementation of the model.

After the dataset was chosen, it was checked for any errors or missing values. The data was then cleaned using various preprocessing methods which included replacing the missing values. Classification was then performed on the data using OneR, J48 and Naïve-Bayes algorithms to find any interesting patterns. Separate tests were run with different parameters to check for the highest accuracy rates of predicting the data. Visual tree structures were used in representing data to make it easier for the user to understand it.

The results of the tests varied slightly. Even though, Naïve-Bayes had a lower accuracy rate as compared to J48, the voting patterns were much more clear when this algorithm was used. The J48 provided the highest accuracy rate in predicting the data.

The experiments performed were an excellent way of gaining an understanding of the data, as well as learning the intricacies of the algorithms used. The result showed an interesting pattern that can be extremely beneficial in using the same methods for a comparative study of similar datasets. The congressional voting patterns can be easily compared by these methods and changes in policy and agenda of the political parties can be identified.

Introduction:

The main objective of this report is to explore the dataset for any information that could be extracted related to the congressional voting patterns in the United States. Upon studying the dataset, several questions arose on the agenda of the two political parties (Democrats and Republicans) that were mentioned in the dataset. What are the priorities of these political parties? What are the issues they are voting in favor of? What are the issues they are voting against for? Which issues form the key agenda for them? How many individual members of the respective parties are voting positively or negatively on the said agendas? And finally, can this dataset be useful in analyzing the congressional voting patterns over long periods of time?

The purpose of this report is to use different methods of preprocessing and data mining techniques to analyze the results. Several evaluation criteria will be used to dissect the whole process and compare the patterns. The expected results should show intensive mining to be more accurate with a high processing time. Simplicity and Interpretability of the final output will be evaluated.

The mining task, primarily, comprises of two focus areas.

- To analyze the overall voting pattern of a political party in relation to a set agenda.
- To analyze the data of individual members within each political party who agree or disagree with the set agenda.

This will allow us to have a comprehensive analysis of the key issues that define a political party in the United States political spectrum. It will also aid the general populace to gain a better understanding of the agenda of different political parties. Furthermore, it can be used as a comparative study to analyze the changing behavior of the political parties in course of time.

Related Work:

One of the tests will involve classifying the data using the decision tree algorithm. The accuracy of its predictions will be measured to determine its usefulness in being used to analyze future voting patterns.

Decision tree algorithms are very useful in visualizing the data. Rahman & Hasan (2011) document their discoveries while using decision trees to classify hospital patients according to their critical condition, based on the patient's current physical condition and medical history. It was the aim of their research to aid hospital staff in classifying patients quickly due to the busy nature of hospital admittance and the shortage of physicians.

Initially in their report, the researches discuss the importance of pre-processing which later is reaffirmed as their first experiment results required them to reprocess their data. This occurred several times until they were able to produce an adequately sized tree that was not overfitted due to the high-dimensionality of their data. They started with 40 attributes and were able to reduce this down to 22 by analyzing the information gain values of all the attributes.

The number of attributes in the given dataset is not large (attributes – 16 + class name = 17). Hence a decision tree algorithm like J48 can help us visually understand the correlation between the attributes and the class chosen.

Classification predicts categories from class labels. The data is then classified to construct models based on training sets and the values of the attributes. The J48 algorithm was created from an earlier classification algorithm known as C4.5 which was developed by John Ross Quinlan. Within the J48 algorithm, as mentioned above, we can create a decision tree which allows for easy viewing of the data. The J48 algorithm also allows for many options of pruning. (Witten IH, 2006)

In addition to this, other algorithms like the Naïve Bayes and OneR are used.

Method:

Description of data:

The “Congressional Voting Records” dataset from the UCI Machine Learning Repository was chosen for this research as it has a sizable number of records which can be used for mining purposes. This data set contains information about how each of the U.S. House of Representatives Congressmen voted on the 16 key notes in the year 1984. One instance represents the voting history of one congressmen and her/his party affiliation. For classification, the party affiliation was used as class attribute.

```
Number of Instances: 435 (267 democrats, 168 republicans)

Number of Attributes: 16 + class name = 17 (all Boolean valued)

Attribute Information:
   1. Class Name: 2 (democrat, republican)
   2. handicapped-infants: 2 (y,n)
   3. water-project-cost-sharing: 2 (y,n)
   4. adoption-of-the-budget-resolution: 2 (y,n)
   5. physician-fee-freeze: 2 (y,n)
   6. el-salvador-aid: 2 (y,n)
   7. religious-groups-in-schools: 2 (y,n)
   8. anti-satellite-test-ban: 2 (y,n)
   9. aid-to-nicaraguan-contras: 2 (y,n)
  10. mx-missile: 2 (y,n)
  11. immigration: 2 (y,n)
  12. synfuels-corporation-cutback: 2 (y,n)
  13. education-spending: 2 (y,n)
  14. superfund-right-to-sue: 2 (y,n)
  15. crime: 2 (y,n)
  16. duty-free-exports: 2 (y,n)
  17. export-administration-act-south-africa: 2 (y,n)
```

The values were all Boolean in nature and all the attributes seemed relevant to analyze the voting patterns. There were missing values as well, a total of 288, represented by the symbol “?”. Furthermore, the missing value percentages of all the attributes was quite small (below 10%). Only the attribute, “export-administration-from-south-africa”, had missing values of 24%. This has a margin of error of approximately 1 in 4 which is not significantly high. Hence, none of the attributes were needed to be removed for the mining tasks.

Website: http://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records

Data mining Area Justification:

Data choice:

The Congressional Vote dataset was chosen as it can provide several patterns that are related to voting by two different parties on multiple policy matters. This data can be used to analyze the policies and agenda that a political party has on key issues related to the people. Furthermore, all the attributes in the dataset were Boolean in nature and it can help us to easily compare the results in an effective manner.

Chosen method used for analysis:

Classification was chosen as a method to analyze the data. It gave the ability to view the false positive (FP), true positive (TP), true negative (TN) and false negative (FN), the confusion matrices and various other statistical measurements as well as viewing the data in a tree format. Most importantly, the tree format can be an effective visual tool to analyze the voting patterns in the dataset. Furthermore, we can check the accuracy results of the classification algorithm. Different algorithms can be compared to get the best accuracy result for the given dataset. The timeliness of the whole process is also a guiding factor in choosing the best classification algorithm.

Chosen Algorithms:

ZeroR:

ZeroR algorithm is used to check the baseline accuracy of the data.

OneR:

OneR, short for "One Rule", is a simple, yet accurate, classification algorithm that generates one rule for each predictor in the data, then selects the rule with the smallest total error as its "one rule". It is the simplest form of analyzing the data and is used as a starting point before going to more complex analysis.

J48:

J48 was chosen as it was the primary classifying algorithm to which most other algorithms have made modifications and derived from. Also, the decision tree structure will be an easy to use interface to analyze the data.

Naïve Bayes:

Naïve Bayes was chosen as it is easy to implement, needs only a small amount of training data and can handle large numbers of attributes while still producing good results.

Pre-processing Details:

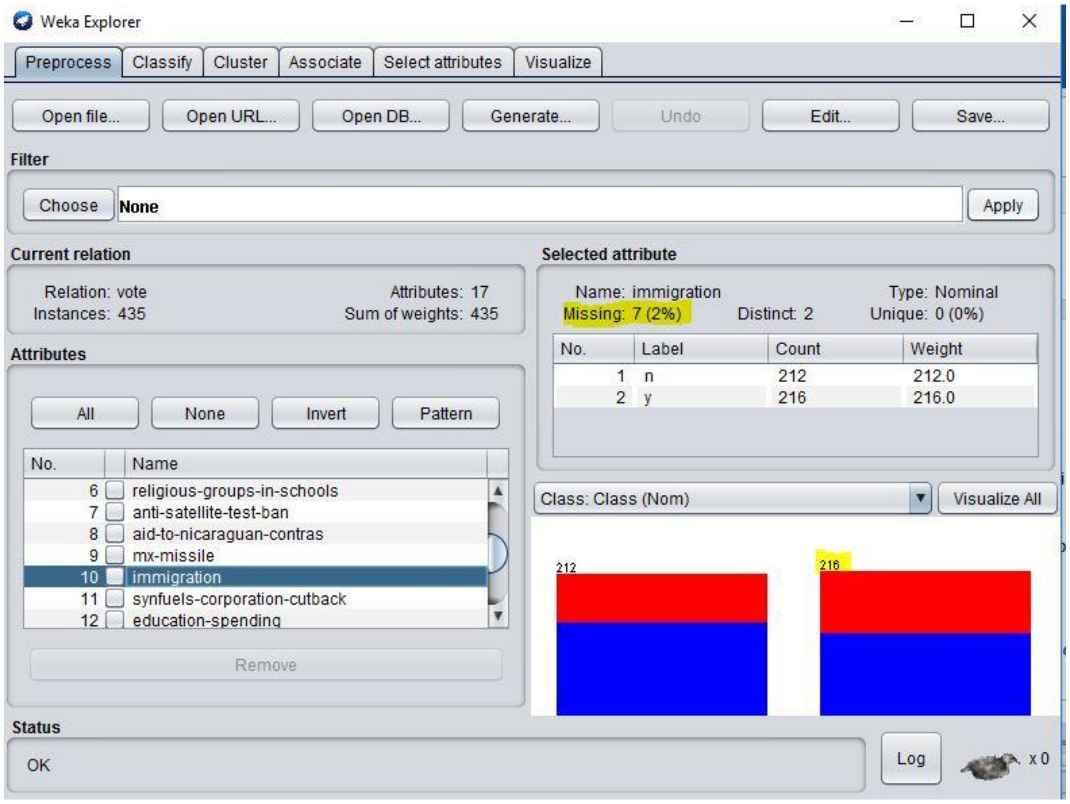
After the data was collected, it was run through Weka (open source machine learning software). This was done before the replacing the "?" values, to attempt to understand how the Weka program worked. The text file was incompatible and the software could not run it. The data was then transformed into a csv file (using excel) and the headings were manually added and later converted to an ARFF file. For an ARFF file, the Relation Declaration, Attribute Declaration and the data type will appear in the beginning before the actual data.

Weka can now read the file and it was ready for pre-processing. However, the given dataset had missing values (288 in total).

"ReplaceMissingValues" algorithm was used in the Weka to replace all the missing values. Once the algorithm was run, all the missing values for each of the attributes were replaced and it showed a 100% accuracy.

The figures below illustrate how the missing values were replaced under the attribute "Immigration" with the help of Weka.

Figure-1: "Immigration" attribute with 7 missing values (2%) as seen before using the "ReplaceMissingValues" algorithm



Filter: None

Current relation
Relation: vote
Instances: 435
Attributes: 17
Sum of weights: 435

Selected attribute
Name: immigration
Type: Nominal
Missing: 7 (2%)
Distinct: 2
Unique: 0 (0%)

No.	Label	Count	Weight
1	n	212	212.0
2	y	216	216.0

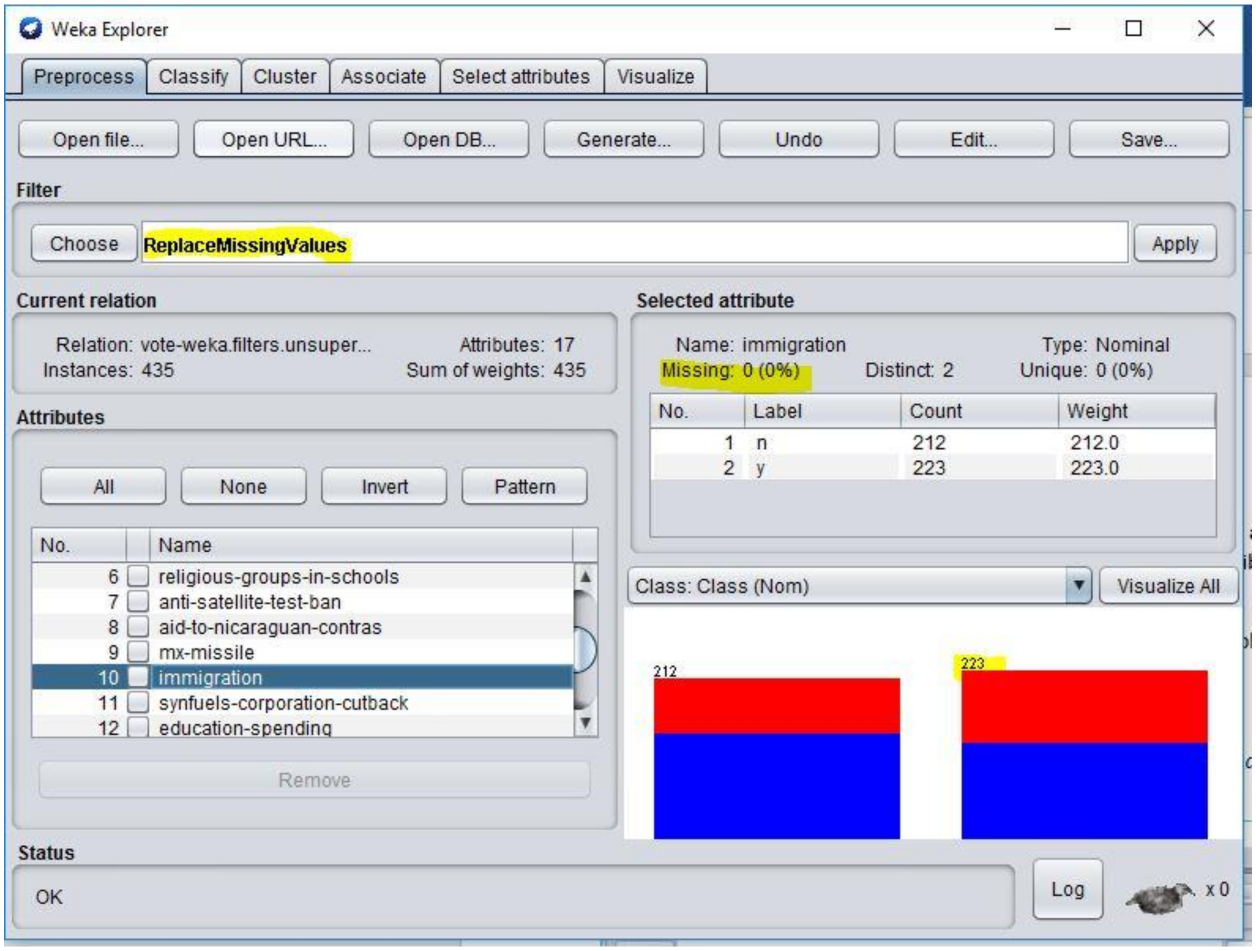
Attributes

No.	Name
6	religious-groups-in-schools
7	anti-satellite-test-ban
8	aid-to-nicaraguan-contras
9	mx-missile
10	immigration
11	synfuels-corporation-cutback
12	education-spending

Class: Class (Nom)

Status
OK

Figure-2: "Immigration" attribute with no missing values (0%) as seen after using the "ReplaceMissingValues" algorithm



Filter

ReplaceMissingValues

Current relation

Relation: vote-weka.filters.unsuper... Attributes: 17
Instances: 435 Sum of weights: 435

Selected attribute

Name: immigration Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	n	212	212.0
2	y	223	223.0

Attributes

No.		Name
6		religious-groups-in-schools
7		anti-satellite-test-ban
8		aid-to-nicaraguan-contras
9		mx-missile
10		immigration
11		synfuels-corporation-cutback
12		education-spending

Class: Class (Nom)

Status

OK

The data types are Boolean in nature and contains no numeric data. The missing values were all filled after running the algorithm. Hence no further preprocessing was felt necessary and the data was ready to be used for analysis via classification.

Classification:

Data mining is an experimental science. There is no universally best learning algorithm. The success of a machine learning method depends on the domain. The aim of the classification testing is to measure the prediction accuracy and the processing time of each test to determine the effectiveness of the algorithm on the selected dataset. Several evaluation criteria are looked at below. These evaluation techniques were used on the training data as well as on the test data (using percentage split in Weka) to understand the relevant patterns.

Evaluation methods:

• Predictive (Classification) accuracy: This refers to the ability of the model to correctly predict the class label of new or previously unseen data.

• Accuracy = % of testing set examples correctly classified by the classifier

• Speed: This refers to the computation costs involved in generating and using the model

• Robustness: This is the ability of the model to make correct predictions given noisy data or data with missing values

•Scalability: This refers to the ability to construct the model efficiently given large amount of data

• Interpretability: This refers to the level of understanding and insight that is provided by the model

• Simplicity: • decision tree size • rule compactness • Domain-dependent quality indicators

Before we begin checking the accuracy of the results obtained by running several algorithms, it is good practice to determine the Baseline Accuracy of the dataset. This can be done in Weka by using the ZeroR classifier.

Algorithms and Results:

Chosen Algorithm – ZeroR:

ZeroR is the simplest classification method which relies on the target and ignores all predictors. ZeroR classifier simply predicts the majority category (class). Although there is no predictability power in ZeroR, the baseline performance can be used as a benchmark for other classification methods.

The correctly classified instances are predicted as 61.3793% accurate and the class value predicted is the democrat.

ZeroR results:

```
=== Run information ===

Scheme:      weka.classifiers.rules.ZeroR
Relation:    vote
Instances:   435
Attributes:  17
```

```
              handicapped-infants
              water-project-cost-sharing
              adoption-of-the-budget-resolution
              physician-fee-freeze
              el-salvador-aid
              religious-groups-in-schools
              anti-satellite-test-ban
              aid-to-nicaraguan-contras
              mx-missile
              immigration
              synfuels-corporation-cutback
              education-spending
              superfund-right-to-sue
              crime
              duty-free-exports
              export-administration-act-south-africa
              Class
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

ZeroR predicts class value: democrat

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.63 seconds

=== Summary ===

Correctly Classified Instances         267               61.3793 %
Incorrectly Classified Instances       168               38.6207 %
Kappa statistic                          0
Mean absolute error                      0.4742
Root mean squared error                  0.4869
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              435

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                  1.000    1.000    0.614      1.000    0.761      0.000    0.500     0.614     democrat
                  0.000    0.000    0.000      0.000    0.000      0.000    0.500     0.386     republican
Weighted Avg.     0.614    0.614    0.377      0.614    0.467      0.000    0.500     0.526
```

```
=== Confusion Matrix ===

   a   b   <-- classified as
 267   0 |   a = democrat
 168   0 |   b = republican
```

This baseline accuracy can be compared with other accuracy results that are obtained after running the several different classifier algorithms like J48 or Naïve Bayes. If these accuracy results are found to be less than the baseline accuracy of 61.3793%, it would imply that the attributes in the dataset are not informative and may need further cleaning. This kind of method will stop us from blindly applying Weka on every dataset.

Chosen Algorithm – OneR:

Before running any complex algorithm, we try to test our data with the simplest of all the algorithms, OneR. There are several different kinds of a simple data structure. It might be that only one attribute in the dataset does all the work. We can check if the dataset gives entirely accurate results even while working on this single attribute. Or, it might be that all the attributes contribute equally and independently (as is the case with Naïve Bayes algorithm which is covered later in the report). There is no universally best learning algorithm. The success of machine learning depends on the domain chosen.

OneR, short for "One Rule", is a simple, yet accurate, classification algorithm that generates one rule for each predictor in the data, then selects the rule with the smallest total error as its "one rule". It is often seen that OneR produces rules only slightly less accurate than state-of-the-art classification algorithms while producing rules that are simple for humans to interpret.

OneR results:

```
=== Run information ===

Test mode:    evaluate on training data

=== Classifier model (full training set) ===

physician-fee-freeze:
	n	-> democrat
	y	-> republican
(416/435 instances correct)


Time taken to build model: 0.02 seconds
```

```
=== Evaluation on training set ===

Time taken to test model on training data: 0.01 seconds

=== Summary ===

Correctly Classified Instances        416               95.6322 %
Incorrectly Classified Instances       19                4.3678 %
Kappa statistic                         0.9088
Mean absolute error                     0.0437
Root mean squared error                 0.209
Relative absolute error                 9.2105 %
Root relative squared error            42.9251 %
Total Number of Instances             435

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.948    0.030    0.981      0.948    0.964      0.910    0.959     0.961     democrat
                 0.970    0.052    0.921      0.970    0.945      0.910    0.959     0.905     republican
Weighted Avg.    0.956    0.039    0.958      0.956    0.957      0.910    0.959     0.940

=== Confusion Matrix ===

   a   b   <-- classified as
 253  14 |   a = democrat
   5 163 |   b = republican
```

Analysis of OneR results:

We can see the accuracy of the correctly classified instances is 95.6322%. This indicates a very high level of accuracy in classifying the data.

The time taken in building the model is 0.02 seconds and the time taken to test the model is 0.01 seconds. This indicates a relatively high performance in the time taken to process the data.

We look at the rule identified by the classifier. The classifier chose the attribute "physician-fee-freeze" as the base for its calculations. If it's a "n" choose "democrat", and if it's a "y" choose "republican". We get 416 of the 435 instances correct on the training set.

Also, the accuracy rate of the OneR is much higher than the ZeroR accuracy which is only 61.3793%. This indicates that OneR classifier is working well for the chosen dataset.

The True Positive(TP) and the False Positive(FP) values can also be analyzed from the results. The democrats are classified with a TP rate of 94.8% and the republicans with 97%.

This shows a high rate of success in performing the classification.

Now, we look at the Decision Tree Algorithm that can be used to find even more interesting patterns in the dataset.

Decision Tree Algorithm:

Decision trees provide a set of rules that one can apply to a new dataset to predict the outcome. They are a tree like structure where each branch node represents a choice between alternatives and each leaf represents a classification. Think of the branch node as the question, the branch as the rule and the leaf as the statement. The number of levels or branches on the tree can be user-defined which affects the outcomes of the tree (Aitkenhead, 2008). Incorrectly setting the depth of the tree can lead to under or overfitting, which in the case of overfitting, leads the model to be too rigid to fit any other data sample. Decision trees can represent the acquired knowledge in a visual form and are easy to interpret if they are not overly large which can lead to overfitting. They are also good dealing with noisy or incomplete data and can handle both continuous and discrete data.

In the case of the chosen dataset, decision tree is the easiest way to visualize the relation between the class and the attributes. This will be illustrated with the help of pruned and unpruned trees. To do that we use one of the most popular decision tree algorithms, J48, for classifying the dataset. We use the classifier on three datasets – the training dataset and two test datasets (percentage split of 66% and 90% using Weka). By using this way, we can analyze if the classifier can be applied to any other similar real world scenario.

Chosen Algorithm - J48:

J48 is one of the most extensively used and popular decision tree algorithms in Weka. Its main advantage is the short amount of time in which the process is done and its effectiveness in dealing with noisy data (our chosen data is clean and has no missing values now). The accuracy results of J48 classifier are usually on the higher side in comparison with other classifiers.

We use the training dataset and test datasets (percentage split of 66% and 90%) and run the classifier to analyze the results.

Training dataset Results (J48):

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     vote
Instances:    435
Attributes:   17
              handicapped-infants
              water-project-cost-sharing
              adoption-of-the-budget-resolution
              physician-fee-freeze
              el-salvador-aid
              religious-groups-in-schools
              anti-satellite-test-ban
              aid-to-nicaraguan-contras
              mx-missile
              immigration
              synfuels-corporation-cutback
              education-spending
              superfund-right-to-sue
              crime
              duty-free-exports
              export-administration-act-south-africa
              Class
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

J48 pruned tree
------------------

physician-fee-freeze = n: democrat (253.41/3.75)
physician-fee-freeze = y
|   synfuels-corporation-cutback = n: republican (145.71/4.0)
|   synfuels-corporation-cutback = y
|   |   mx-missile = n
|   |   |   adoption-of-the-budget-resolution = n: republican (22.61/3.32)
|   |   |   adoption-of-the-budget-resolution = y
|   |   |   |   anti-satellite-test-ban = n: democrat (5.04/0.02)
|   |   |   |   anti-satellite-test-ban = y: republican (2.21)
|   |   mx-missile = y: democrat (6.03/1.03)

Number of Leaves  :     6

Size of the tree :      11


Time taken to build model: 0.02 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.02 seconds

=== Summary ===
Correctly Classified Instances       423               97.2414 %
Incorrectly Classified Instances      12                2.7586 %
Kappa statistic                        0.9418
Mean absolute error                    0.0519
Root mean squared error                0.1506
Relative absolute error               10.9481 %
Root relative squared error           30.9353 %
Total Number of Instances            435

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.978    0.036    0.978      0.978    0.978      0.942    0.986     0.987     democrat
                 0.964    0.022    0.964      0.964    0.964      0.942    0.986     0.970     republican
Weighted Avg.    0.972    0.031    0.972      0.972    0.972      0.942    0.986     0.981

=== Confusion Matrix ===

   a   b   <-- classified as
 261   6 |   a = democrat
   6 162 |   b = republican
```

Test dataset Results (Percentage split of 66%)- J48:

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     vote-weka.filters.unsupervised.attribute.ReplaceMissingValues
Instances:    435
Attributes:   17
              handicapped-infants
              water-project-cost-sharing
              adoption-of-the-budget-resolution
              physician-fee-freeze
              el-salvador-aid
              religious-groups-in-schools
              anti-satellite-test-ban
              aid-to-nicaraguan-contras
              mx-missile
              immigration
              synfuels-corporation-cutback
              education-spending
              superfund-right-to-sue
              crime
              duty-free-exports
              export-administration-act-south-africa
              Class
```

```
Test mode:    split 66.0% train, remainder test

=== Classifier model (full training set) ===

J48 pruned tree
------------------

physician-fee-freeze = n: democrat (258.0/5.0)
physician-fee-freeze = y
|   synfuels-corporation-cutback = n: republican (145.0/3.0)
|   synfuels-corporation-cutback = y
|   |   mx-missile = n
|   |   |   adoption-of-the-budget-resolution = n: republican (21.0/3.0)
|   |   |   adoption-of-the-budget-resolution = y
|   |   |   |   water-project-cost-sharing = n: republican (2.0)
|   |   |   |   water-project-cost-sharing = y: democrat (4.0)
|   |   mx-missile = y: democrat (5.0/1.0)

Number of Leaves  :       6

Size of the tree :        11


Time taken to build model: 0,02 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.01 seconds

=== Summary ===

Correctly Classified Instances         143               96.6216 %
Incorrectly Classified Instances         5                3.3784 %
Kappa statistic                          0.9311
Mean absolute error                      0.0807
Root mean squared error                  0.1801
Relative absolute error                 16.845  %
Root relative squared error             36.3306 %
Total Number of Instances              148

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.953    0.016    0.988      0.953    0.970      0.932    0.969     0.969     democrat
                 0.984    0.047    0.938      0.984    0.961      0.932    0.969     0.930     republican
Weighted Avg.    0.966    0.029    0.967      0.966    0.966      0.932    0.969     0.953
```

```
=== Confusion Matrix ===

  a  b   <-- classified as
 82  4 |  a = democrat
  1 61 |  b = republican
```

Test Dataset Results (Percentage split of 90%) – J48:

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     vote
Instances:    435
Attributes:   17
              handicapped-infants
              water-project-cost-sharing
              adoption-of-the-budget-resolution
              physician-fee-freeze
              el-salvador-aid
              religious-groups-in-schools
              anti-satellite-test-ban
              aid-to-nicaraguan-contras
              mx-missile
              immigration
              synfuels-corporation-cutback
              education-spending
              superfund-right-to-sue
              crime
              duty-free-exports
              export-administration-act-south-africa
              Class
Test mode:    split 90.0% train, remainder test

=== Classifier model (full training set) ===

J48 pruned tree
------------------

physician-fee-freeze = n: democrat (253.41/3.75)
physician-fee-freeze = y
|   synfuels-corporation-cutback = n: republican (145.71/4.0)
|   synfuels-corporation-cutback = y
|   |   mx-missile = n
|   |   |   adoption-of-the-budget-resolution = n: republican (22.61/3.32)
|   |   |   adoption-of-the-budget-resolution = y
|   |   |   |   anti-satellite-test-ban = n: democrat (5.04/0.02)
|   |   |   |   anti-satellite-test-ban = y: republican (2.21)
|   |   mx-missile = y: democrat (6.03/1.03)

Number of Leaves  :       6

Size of the tree :            11


Time taken to build model: 0.02 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.01 seconds

=== Summary ===

Correctly Classified Instances          41              95.3488 %
Incorrectly Classified Instances         2               4.6512 %
Kappa statistic                          0.8844
Mean absolute error                      0.0797
Root mean squared error                  0.2194
Relative absolute error                 17.5041 %
Root relative squared error             47.2684 %
Total Number of Instances               43

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                  0.968    0.083    0.968      0.968    0.968      0.884    0.953     0.966     democrat
                  0.917    0.032    0.917      0.917    0.917      0.884    0.953     0.927     republican
Weighted Avg.     0.953    0.069    0.953      0.953    0.953      0.884    0.953     0.955

=== Confusion Matrix ===

  a  b   <-- classified as
 30  1 |  a = democrat
  1 11 |  b = republican
```

We can compare the results in the table below:

Dataset	Correctly Classified instances (Accuracy)	Time taken to build model	Time taken to test the model
Training dataset	97.2414%	0.02 seconds	0.02
Test Dataset (Percentage split 66%)	96.6216%	0.02 seconds	0.01
Test Dataset (Percentage split 90%)	95.3488%	0.02 seconds	0.01

Analysis of the results:

The above table clearly indicates the accuracy of the J48 algorithm with the chosen dataset. All the results turned out to have high accuracies. The test dataset with 90% percentage split shows the least accuracy but not with a wide margin. This margin may be a problem for very large datasets, but for the current dataset this is an acceptable range. We can also see that all the accuracy rates are much higher than the baseline accuracy of 61.3783% (calculated by using the ZeroR classifier). If any of the values were lower than the baseline accuracy, we can assume that the attributes of the dataset are not entirely informative and may require more cleaning and preprocessing.

The high accuracy results in all the tests indicate that we can deploy the classifier in some real situation to make predictions on fresh data samples.

The True Positive (TP) rate of the classification shows up on the higher side in all the test results. This shows the number of correct instances correctly classified. This is a good indication of the accuracy of the end-result.

The False Positive (FP) rate refers to the instances falsely classified as a given class.

By analyzing the table below showing the TP and the FP rates, we can clearly see that the True Positives show a high percentage and the False positives have very low values. This indicates that a high percentage of the instances are correctly detected and classified by the algorithm.

True Positive (TP) rate:

Class	Training Dataset	Testing dataset (66% split)	Testing dataset (90% split)
Democrat	0.978	0.953	0.968
Republican	0.964	0.984	0.917

False Positive (FP) rate:

Class	Training Dataset	Testing dataset (66% split)	Testing dataset (90% split)
Democrat	0.036	0.016	0.083
Republican	0.022	0.017	0.032

The time taken to build the model is also a crucial factor in determining the performance of the classifier. This is extremely important in dealing with very large datasets. The table shows minor variations in the time taken to build the models. One interesting observation is with time taken to test the model for the training dataset is 0.02 seconds whereas it shows as 0.01 seconds for the remaining two tests. This may be because the training data is larger in size as compared to the test datasets (which are only a fraction of the main dataset – Testing dataset with percentage split of 66% means 66% is used as training data and the remaining 34% is used as the test data. the same can applied to the third set of data with a percentage split of 90%. In that case the test data is only 10% and much smaller in size).

To obtain high performance and speed of processing, Time is considered as an important factor in determining the efficiency of using the classifier, especially when used with very large datasets.

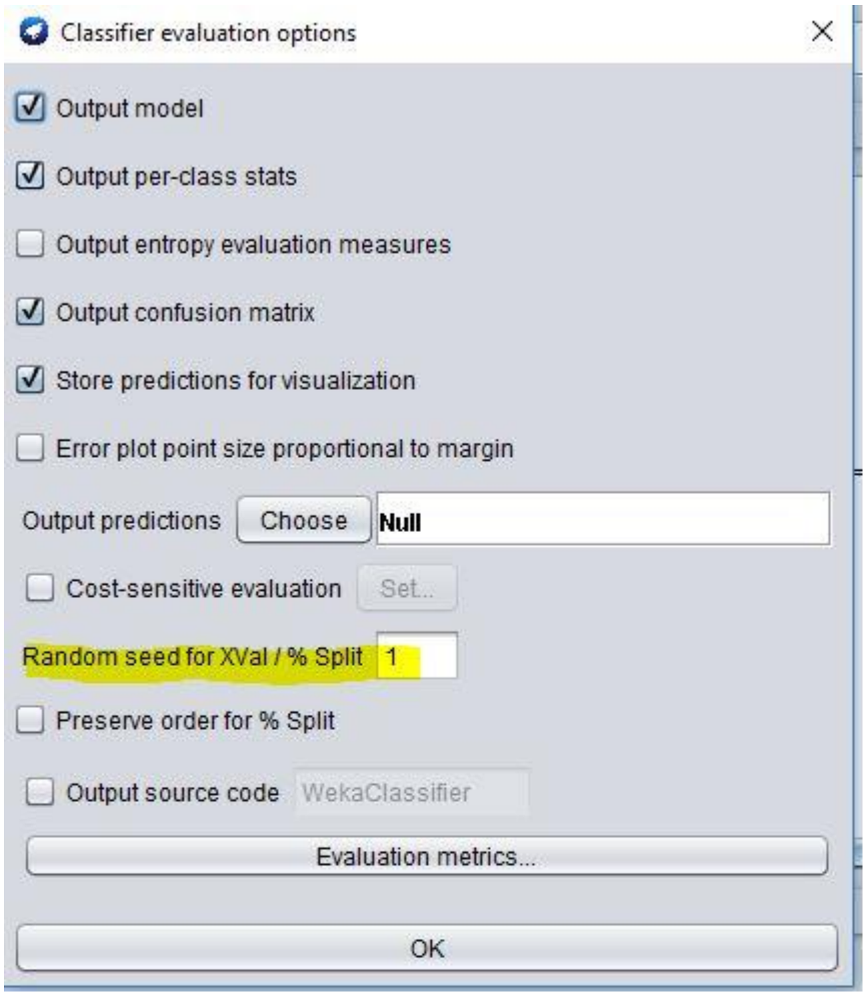
The accuracy of the dataset can also be attributed to the non-missing values in the chosen dataset. As data was already cleaned of any missing values during the preprocessing stage, it is rendered with more accuracy.

Repeated Training and Testing:

An interesting observation can be made by looking at the accuracy results of the previous tests. All the instances were showing a very high accuracy rate in every test conducted. This may sometimes be a misleadingly high optimistic figure.

We are going to repeatedly train and test with the percentage split of 90% on the dataset. We change the number of random seeds in the options menu as shown below, and test the data with different number of random seeds from 1 to 10.

Figure 3 – The figure shows the options menu in the classify panel to change the number of random seeds.


Classifier evaluation options

[✓] Output model

[✓] Output per-class stats

[ ] Output entropy evaluation measures

[✓] Output confusion matrix

[✓] Store predictions for visualization

[ ] Error plot point size proportional to margin

Output predictions: Choose — Null

[ ] Cost-sensitive evaluation — Set...

Random seed for XVal / % Split: 1

[ ] Preserve order for % Split

[ ] Output source code — WekaClassifier

Evaluation metrics...

OK

The accuracy rate of the correctly classified instances over the 10 runs is as follows:

Algorithm used J48

Percentage Split set to 90%

Initial run with default number of random seeds, that is 1

- Gives an accuracy of 95.35%

Repeated with number of seeds 2,3,4,5,6,7,8,9,10 gives the following results.

Table showing the correctly classified instance rate after repeated tests with different values of random-number seeds:

Number of Random seeds	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Accuracy-Correctly classified instances	0.953	0.976	0.992	0.998	1.000	1.000	0.967	0.954	0.967	0.967

Given these set of experimental results, we can calculate the mean and the standard deviation. The sample mean is the sum of all these accuracy rates divided by the number 10. We calculate the sample variance from the mean. We take the deviation from the mean, we subtract the mean from each of these numbers, we square that, add them up and we divide not by n, but n-1. The reason for n-1 is because we have calculated the mean from this sample. When the mean is calculated from this sample, we need to divide it by n-1, leading to a slightly larger variance estimate than if we were to divide by n. We take the square root of that value and we get the standard deviation.

Sample mean: $= (\Sigma x_i) / n$.

Variance $$s^2 = \frac{\sum (X - \overline{X})^2}{N-1}$$

Standard Deviation = s

Sample mean = 0.977, which means about 97.7%. Standard Deviation (s)= 0.018

This gives us a better estimate than the actual 95.3% which we started with. It can be considered a more reliable estimate of the accuracy rate of the correctly classified instances in this test dataset.

With this repeated testing, we can see the real performance of the J48 algorithm on the testing dataset. The basic assumption is that the training and test datasets are sampled independently from an infinite population, and we should expect a slight variation in results. In this case, the

the variation has a margin of error of about 2 to 3 % with a 97.7% accuracy. With the help of all these tests, it can be concluded that the chosen dataset has a very high accuracy rate in correctly classifying the instances by suing these algorithms. We can also estimate the variation in results by setting the random-number seed and repeating the experiment.

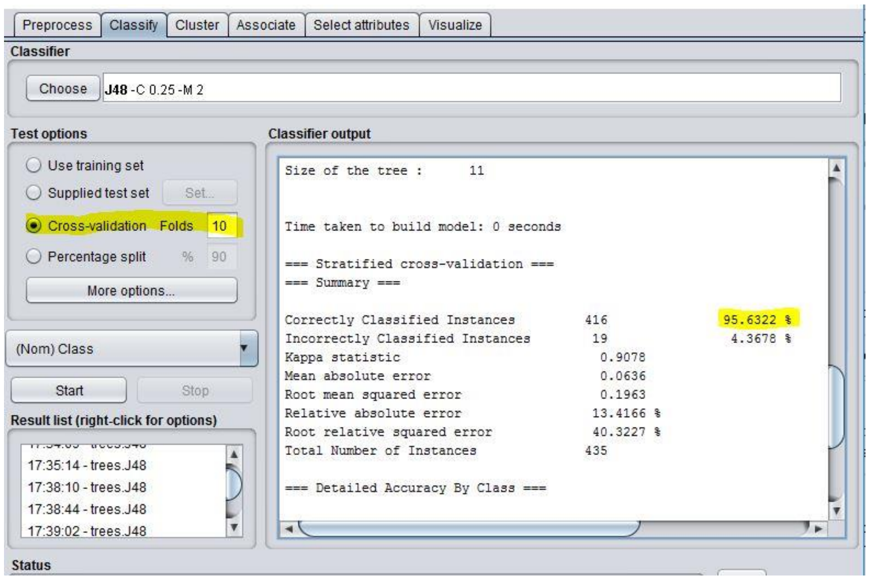
Cross-Validation:

There are other methods like cross-validation which can be used to improve upon the repeated holdout method which we just used with the random-number seeds. Cross validation is a systematic way of using repeated holdout that improves upon it by reducing the variance of the estimate. Stratified cross validation reduces the variance even further. Weka does Stratified cross validation by default. With 10-fold cross validation, Weka invokes the learning algorithm 11 times, one for each fold of the cross validation and then a final time on the entire dataset.

A practical rule of thumb is, if we have a large dataset, we can use percentage split and evaluate it just once. Otherwise, if we don't have too much, we should use stratified 10-fold cross validation. Large dataset usually depends on the number of classes in the dataset. The chosen dataset is not as large and cross-validation can be a reliable way to evaluate it.

Cross validation on the dataset (using the J48 classifier) gives an accuracy rate of 95.6322% for the correctly classified instances. It is pretty high, and in line with the accuracy rates we are getting with all the tests.

Figure-4 Cross validation results:



Classifier: J48 -C 0.25 -M 2

Test options:
- Use training set
- Supplied test set
- [✓ Cross-validation Folds 10]
- Percentage split % 90

(Nom) Class

Result list (right-click for options):
- 17:35:14 - trees.J48
- 17:38:10 - trees.J48
- 17:38:44 - trees.J48
- 17:39:02 - trees.J48

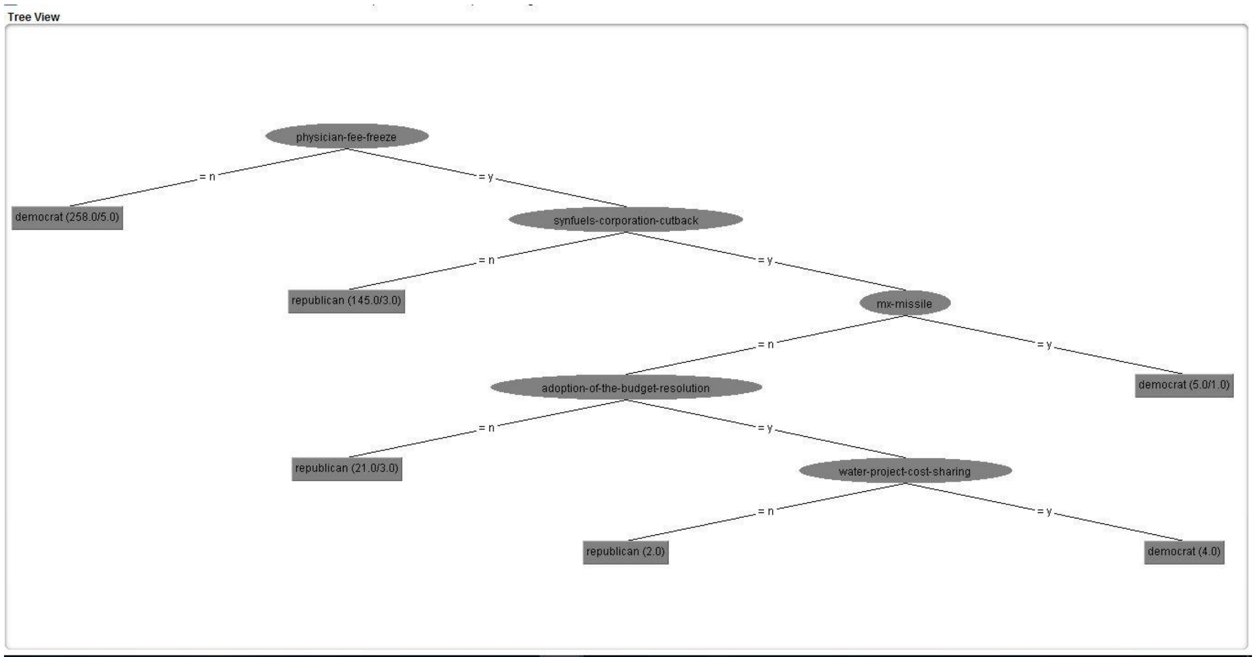
Classifier output:

```
Size of the tree :      11


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         416               95.6322 %
Incorrectly Classified Instances        19                4.3678 %
Kappa statistic                          0.9078
Mean absolute error                      0.0636
Root mean squared error                  0.1963
Relative absolute error                 13.4166 %
Root relative squared error             40.3227 %
Total Number of Instances              435

=== Detailed Accuracy By Class ===
```

J48 Decision Trees (Pruned and Unpruned):

Decision tree algorithms are very useful in their visual representation of data. If we analyze the testing results of the Training dataset, we can see that the J48 automatically runs the algorithm as a pruned tree structure by default. When we use the pruned default setting in the J48, only the attributes which are relevant are displayed. The number of the leaves is shown as 6 and the size of the leaves as 11. The leaves here refer to the internal nodes of the tree.

Figure 5 - The visualization of the tree is shown below:



By this visual representation it can be easily understood which party has voted positively and negatively on which agenda. It is a simple "yes" or "no" structure that can be analyzed easily without having to go through the data. The republicans have voted "no" on most of the issues. We can understand that of the given attributes, the democrats have voted more positively than the republicans (more number of "y"). We can look at this tree and understand the structure of the decision, and see what's important in the data without any pre-requisite knowledge and understanding of the dataset.

In the pruning technique, the algorithm splits at the root node at the attribute from which the most amount of information can be gained. In this case, it is the "physician-fee-freeze" attribute.

We can also test and look at the unpruned tree with all the branches and the leaves. We run the training dataset with the "unpruned" option enabled and get the following results.

Number of leaves: 13

Size of the tree: 25

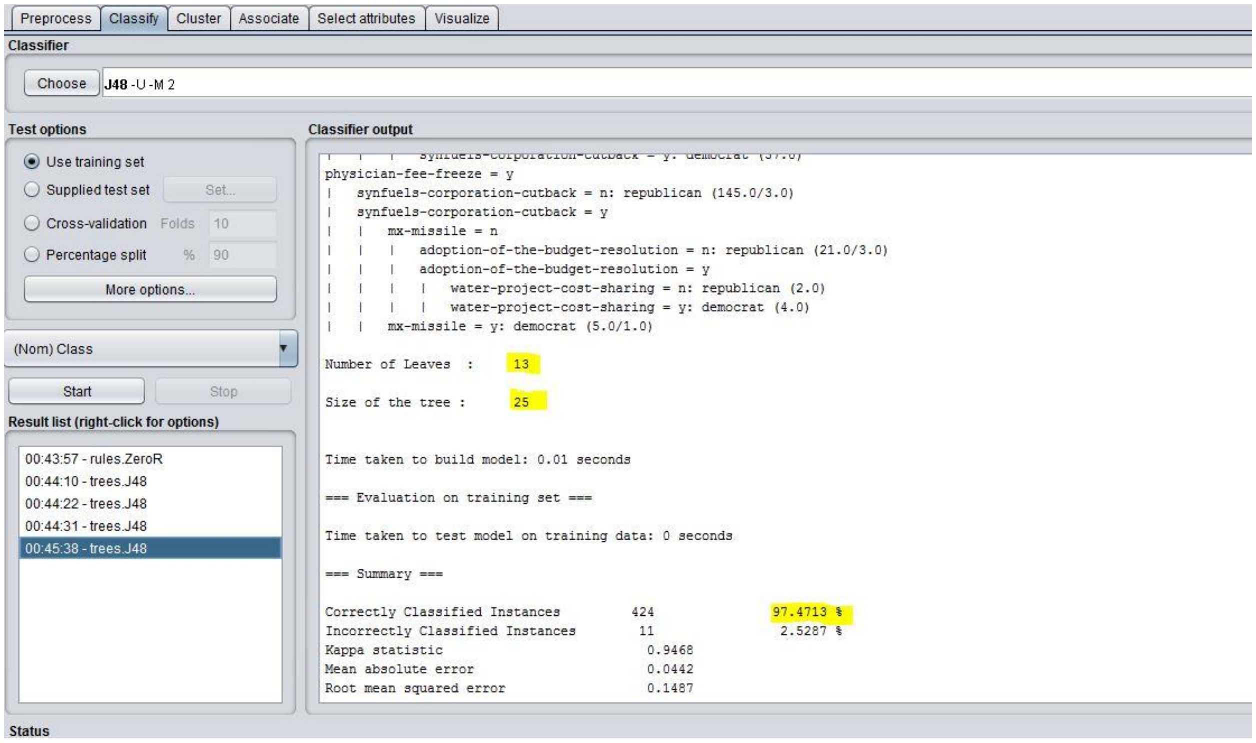
Correctly classified instances: 97.4713%

Even though the accuracy is marginally higher than the pruned tree structure, the number of leaves and the internal nodes will make it a bit more complex to understand the tree structure. However, it is still a very useful representation of finding the agendas the parties have agreed or disagreed with.

The figures below illustrate the test results along with the unpruned tree structure of the training dataset.

In case of large datasets, the unpruned tree structure is not a viable option as it will create a big tree that can be very hard to comprehend.

Training dataset results- unpruned: (Figure 6)



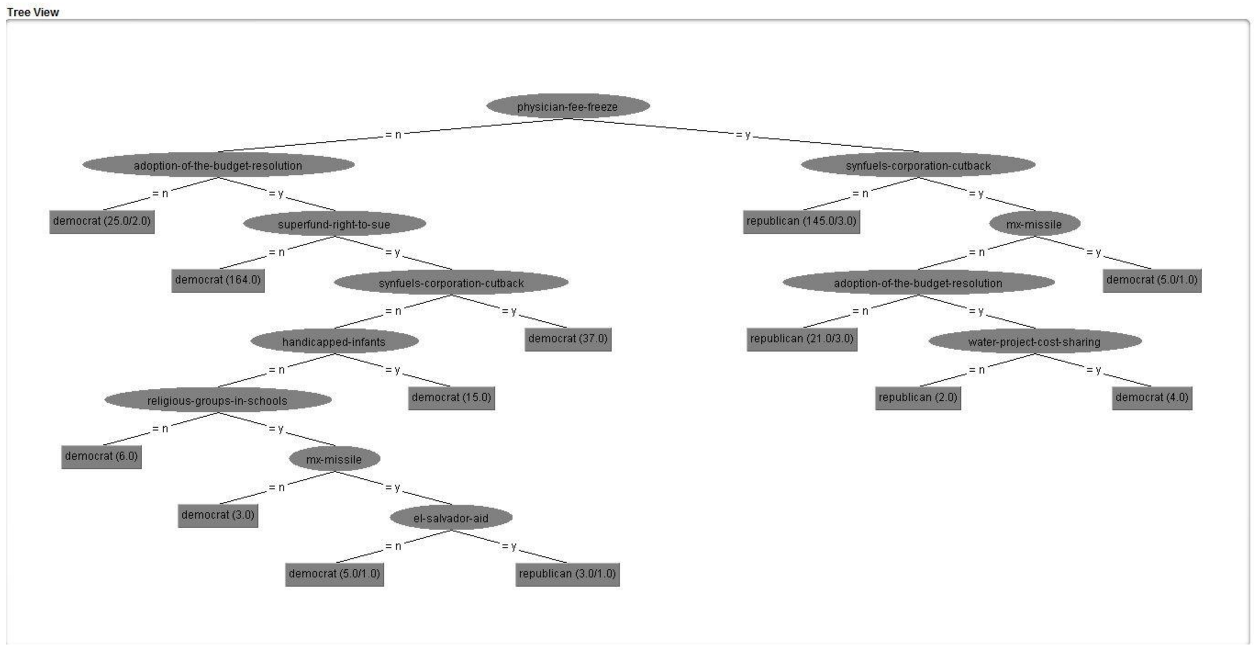
J48 -U -M 2

Test options: (● Use training set) Supplied test set; Cross-validation Folds 10; Percentage split % 90

Result list (right-click for options):
00:43:57 - rules.ZeroR
00:44:10 - trees.J48
00:44:22 - trees.J48
00:44:31 - trees.J48
00:45:38 - trees.J48

```
physician-fee-freeze = y
|   synfuels-corporation-cutback = n: republican (145.0/3.0)
|   synfuels-corporation-cutback = y
|   |   mx-missile = n
|   |   |   adoption-of-the-budget-resolution = n: republican (21.0/3.0)
|   |   |   adoption-of-the-budget-resolution = y
|   |   |   |   water-project-cost-sharing = n: republican (2.0)
|   |   |   |   water-project-cost-sharing = y: democrat (4.0)
|   |   mx-missile = y: democrat (5.0/1.0)

Number of Leaves  :     13

Size of the tree :      25


Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances         424               97.4713 %
Incorrectly Classified Instances        11                2.5287 %
Kappa statistic                          0.9468
Mean absolute error                      0.0442
Root mean squared error                  0.1487
```

Figure 7 - Unpruned tree structure with 13 leaves and 25 internal nodes:



Analysis of the decision tree algorithm:

Decision tree algorithm like J48 can easily interpret the chosen dataset by predicting the accuracy and visualizing the tree structures which help us identify the policies in the agenda that each political party agree or disagree with. In simple terms, it’s a visual tool for the eye to see which party has voted “yes” or “no” on which policy.

Naïve Bayes Algorithm:

Naïve Bayes is a probability based classification method that has comparable performance to decision trees. It is based on applying Bayes theorem with strong independence assumptions which leads to lower computation time but loss of accuracy as dependencies will usually exist between attributes. The advantages of Naïve Bayes classifier is that it is easy to implement, needs only a small amount of training data and can handle large numbers of attributes while still producing good results.

In the Naïve Bayes method, all the attributes contribute equally and independently to the decision. To explain it briefly,

Attributes are:

- Equally important
- Statistically independent (given the class value)
    - i.e., knowing the value of one attribute says nothing about the value of another.

This algorithm is based on the Bayes Theorem, by Thomas Bayes, a British Mathematician (1702 – 1761)

Bayes Theorem:

Probability of event H given evidence E:

$$\Pr[H \mid E] = \frac{\Pr[E \mid H]\Pr[H]}{\Pr[E]}$$

$$\Pr[H \mid E] = \frac{\Pr[E_1 \mid H]\Pr[E_1 \mid H]\ldots\Pr[E_n \mid H]\Pr[H]}{\Pr[E]}$$

Pr[H] is the probability of the event before the evidence is seen.

Pr [H|E] is the probability of event after evidence is seen.

Bayes theorem is about the probability of a hypothesis H given evidence E. In our case, the hypothesis is the class of an instance and the evidence is the attribute values of the instance.

By applying the Naïve Bayes classifier to the data set we get the following results:

Naïve Bayes Results:

```
=== Run information ===

Scheme:      weka.classifiers.bayes.NaiveBayes
Relation:    vote-weka.filters.unsupervised.attribute.ReplaceMissingValues
Instances:   435
Attributes:  17
             handicapped-infants
             water-project-cost-sharing
             adoption-of-the-budget-resolution
             physician-fee-freeze
             el-salvador-aid
             religious-groups-in-schools
             anti-satellite-test-ban
             aid-to-nicaraguan-contras
             mx-missile
             immigration
             synfuels-corporation-cutback
             education-spending
             superfund-right-to-sue
             crime
             duty-free-exports
```

```
            export-administration-act-south-africa
            Class
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

Naive Bayes Classifier

                                    Class
Attribute                        democrat republican
                                   (0.61)    (0.39)
===============================================================
handicapped-infants
  n                                 112.0     138.0
  y                                 157.0      32.0
  [total]                           269.0     170.0

water-project-cost-sharing
  n                                 120.0      74.0
  y                                 149.0      96.0
  [total]                           269.0     170.0

adoption-of-the-budget-resolution
  n                                  30.0     143.0
  y                                 239.0      27.0
  [total]                           269.0     170.0

physician-fee-freeze
  n                                 254.0       6.0
  y                                  15.0     164.0
  [total]                           269.0     170.0

el-salvador-aid
  n                                 201.0       9.0
  y                                  68.0     161.0
  [total]                           269.0     170.0

religious-groups-in-schools
  n                                 136.0      18.0
  y                                 133.0     152.0
  [total]                           269.0     170.0

anti-satellite-test-ban
  n                                  60.0     124.0
  y                                 209.0      46.0
  [total]                           269.0     170.0
```

```
aid-to-nicaraguan-contras
  n                          46.0    134.0
  y                         223.0     36.0
  [total]                   269.0    170.0

mx-missile
  n                          61.0    147.0
  y                         208.0     23.0
  [total]                   269.0    170.0

immigration
  n                         140.0     74.0
  y                         129.0     96.0
  [total]                   269.0    170.0

synfuels-corporation-cutback
  n                         139.0    148.0
  y                         130.0     22.0
  [total]                   269.0    170.0

education-spending
  n                         232.0     34.0
  y                          37.0    136.0
  [total]                   269.0    170.0

superfund-right-to-sue
  n                         180.0     23.0
  y                          89.0    147.0
  [total]                   269.0    170.0

crime
  n                         168.0      4.0
  y                         101.0    166.0
  [total]                   269.0    170.0

duty-free-exports
  n                         108.0    155.0
  y                         161.0     15.0
  [total]                   269.0    170.0

export-administration-act-south-africa
  n                          13.0     51.0
  y                         256.0    119.0
  [total]                   269.0    170.0
```

```
Time taken to build model: 0.1 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.04 seconds

=== Summary ===

Correctly Classified Instances        393               90.3448 %
Incorrectly Classified Instances       42                9.6552 %
Kappa statistic                         0.7994
Mean absolute error                     0.0988
Root mean squared error                 0.2991
Relative absolute error                20.8386 %
Root relative squared error            61.4226 %
Total Number of Instances             435

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                  0.895    0.083    0.945      0.895    0.919      0.801    0.973     0.983     democrat
                  0.917    0.105    0.846      0.917    0.880      0.801    0.973     0.958     republican
Weighted Avg.     0.903    0.092    0.907      0.903    0.904      0.801    0.973     0.974

=== Confusion Matrix ===

   a   b   <-- classified as
 239  28 |   a = democrat
  14 154 |   b = republican
```

Analysis of Naïve-Bayes results:

When we look at the results after running the Naïve Bayes classifier, we can see that the accuracy of the correctly classified instances is at 90.3448%. This has a margin of difference of about 7% when we compare it with the J48 accuracy results. This shows that the J48 is currently the most accurate in predicting the chosen data.

The TP rate for the democrats and the Republicans is shown as 0.895 and 0.917 respectively. The FP rate for the democrats and the republicans is shown as 0.083 and 0.105 respectively.

This indicates that the instances are correctly classified with a high level of accuracy.

However, the significant thing that needs to be observed is the individual attribute values that are displayed using the Naïve Bayes method.

Herein,

- we get a model that shows all the attributes,
    - and then for each of the attribute values, we have got the number of times that attribute value appears.

For example, we look at the first attribute “handicapped-infants”

```
                              Class
Attribute                 democrat republican
                            (0.61)   (0.39)
==============================================================
handicapped-infants
  n                         112.0    138.0
  y                         157.0     32.0
  [total]                   269.0    170.0
```

This gives us a clear count of the actual number of democrats and republicans who voted “yes” or “no” on that policy.

However, care needs to be taken to understand the exact number of “y” and “n”. There is one little and important difference between the results shown and the actual result. The count of all the values is 1 more than the actual count. The reason is Weka adds 1 to all the counts. The reason it does this is to get rid of the zeros. As these calculations are computed by using the Bayes theorem which involves multiplying the probabilities as mentioned above, there will be an error if there is a probability value of 0 for any attribute given “y” or “n”. In this case, the final output will become 0 (anything multiplied by zero becomes zero). This is called a “zero frequency problem”. To avoid this, Weka’s solution is to add 1 to all the counts. This means that all the numbers in the Weka table are 1 more than their actual count.

Actual count of the numbers is:

Number of democrats who agreed and voted “y” on the handicapped-infants policy: 156
Number of democrats who disagreed and voted “n” on the handicapped-infants policy: 111
Number of republicans who agreed and voted “y” on the handicapped-infants policy: 31
Number of republicans who disagreed and voted “n” on the handicapped infants policy: 137

Even though the accuracy of the Naïve Bayes Classifier is slightly lesser than the J48, we can clearly count the actual number of democrats and republicans who agreed or disagreed on the various policies in the agenda. This makes Naïve-Bayes as the most effective classifier to use for the chosen dataset.

Discussion:

Summary Table:

Summary table comparing the results of all the Algorithms used:

Algorithm	Accuracy (correctly classified instances)	Time taken to build data	Time taken to test data
ZeroR	61.3793%	0 seconds	0.63 seconds
OneR	95.6322%	0.02 seconds	0.01 seconds
J48	97.2414%	0.02 seconds	0.02 seconds
Naïve-Bayes	90.3448%	0.1 seconds	0.04 seconds

By comparing all the test results using different classifiers, the following key points are observed:

- J48 algorithm has the highest accuracy rate of predicting the correctly classified instances.
- OneR algorithm shows a comparatively high accuracy rate which indicates its relevance in analyzing the chosen dataset.
- All the three classifiers show much higher accuracy rates when compared to the baseline accuracy of 61.3793% obtained by using the ZeroR algorithm.
- Interestingly, the ZeroR classifier took more time to test the data as compared to the other classifiers.
- The Naïve-Bayes classifier took more time to process the computations than the J48 and OneR classifiers.

Even though, the Naïve -Bayes algorithm has less accuracy than J48 and OneR, it can be considered as the best way to analyze the chosen dataset as it gives the individual count of the number of members in each political party who have voted in favor of or against the policies.

However, the significance of J48 algorithm cannot be ignored as it gives an easy to use visual representation of the data with the help of the decision tree structure.
As mentioned above, Data mining is an experimental science. There is no hard and fast rule that one algorithm is better or more accurate than the other. The viability and efficiency of the chosen algorithm depends on the selected dataset and the patterns we are searching to analyze them. In the current context, Naïve-Bayes gives us a more better understanding of the patterns of the chosen dataset.

Issues:

Several issues arose while implementing the data mining techniques to reveal the patterns from the original datasets.

To begin with, the dataset obtained from the UCI library was not compatible with the Weka software. The software can only run ARFF and CSV files. The text file had to be converted into the ARFF format to be compatible with the machine learning software.

One of the evaluation methods used was to check the time taken to test the data. This time was always not constant. Several runs using the same classifier algorithms repeatedly will give slightly different results. The same algorithm shows up different time values in sequential runs. It was felt that this will create an inconsistent result to analyze. The reasons for this can be many, it also depends on the RAM usage of the laptop which was used to run the software. The application needs to use sufficient memory to turn in consistent results for the final analysis.

The system was also hanging sometimes when the processing was done. This may be a system related or the software related issue.

Understanding the tree structure in the “Unpruned” mode was a challenge as there were multiple number of leaves and internal nodes. It was a time-consuming process that needed multiple checking before understanding the final output.

Conclusion:

After using several data mining techniques to analyze the voting patterns of the two political parties, it was evident that the algorithms used were consistent and effective in creating a model for similar situations. The main-focus areas and the goals set out prior to beginning the data mining task were achieved as the voting patterns were clearly illustrated with the help of the classifiers used. The changing trends in politics can be analyzed with the help of these models. By comparing data from new voting pattern databases with the old ones, we can analyze the shift in the policies of the mainstream political parties over critical issues like Immigration (as is evident in the United States now). These patterns can be used by the general populace to understand the priorities of the various political parties and their agendas.

Future Work:

This report considers only one dataset to analyze the voting patterns of the two political parties. If there is an availability of more datasets pertaining to the voting patterns of several years, more number of conclusions can be drawn by performing similar tests. New patterns will emerge and will be extremely beneficial in determining the shift of a political party in relation to a certain agenda.

References:

Aitkenhead, M. F. (2008). A co-evolving decision tree classification method. Expert Systems with Applications, 34(1), 18-25. doi:10.1016/j.eswa.2006.08.008.

Rahman, R. & Hasan, F. R. (2011). Using and comparing different decision tree classification techniques for mining ICDDR, B Hospital Surveillance data. Expert Systems with Applications, 38(9), 11421-11436. doi.org/10.1016/j.eswa.2011.03.015.

Witten IH, F. E. (2006). Analytics: Decision Tree Induction. Retrieved October 2, 2013, from m.o.n.k.: http://monkpublic.library.illinois.edu/monkmiddleware/public/analytics/decisiontree.html

Advances in Knowledge Discovery and datamining
 - https://pdfs.semanticscholar.org/27a3/1c48354ca47b8a3a5b4902315757802f9341.pdf

An empirical study of the Naïve-Bayes classifier
https://www.researchgate.net/profile/Irina_Rish/publication/228845263_An_Empirical_Study_of_the_Naive_Bayes_Classifier/links/00b7d52dc3ccd8d692000000/An-Empirical-Study-of-the-Naive-Bayes-Classifier.pdf

Rahman, R. & Hasan, F. R. (2011). Using and comparing different decision tree classification techniques for mining ICDDR, B Hospital Surveillance data. Expert Systems with Applications, 38(9), 11421-11436. doi.org/10.1016/j.eswa.2011.03.015.

Tutorialspoint - https://www.tutorialspoint.com/data_mining/dm_classification_prediction.htm

CP3403:Data Mining – Classification [PowerPoint slides]. (n.d.). Retrieved from https://learnjcu.jcu.edu.au