**Veterans Affairs WebPage Data Project Report**

**Team Members:**

**Aneesh Kalla, Anurag Perakalapudi**

**10/21/24**

**Statement of Data Mining Goal:**

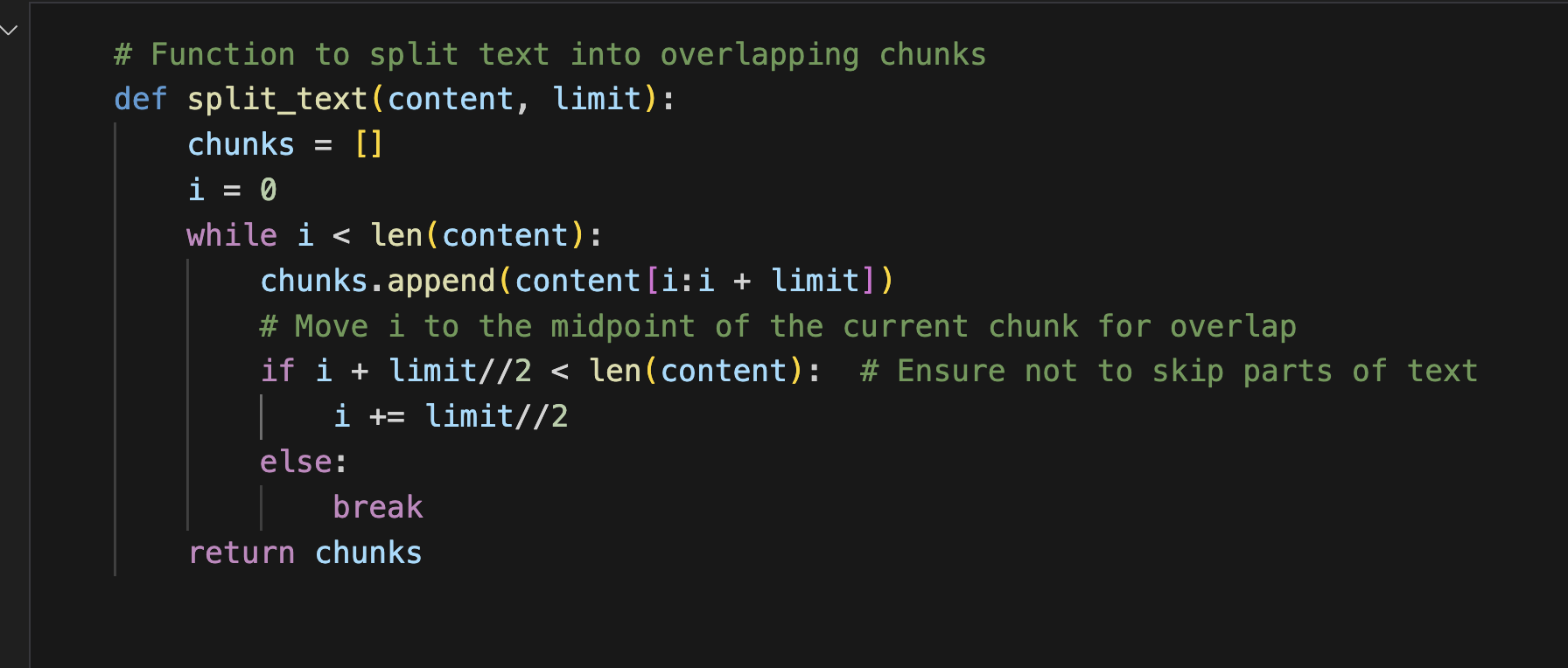
Veterans Affairs (VA) is a government agency that is supposed to process claims and provide benefits to Veterans. However, Veterans often have a very difficult time actually claiming the benefits they deserve, because the process to go through the VA can be difficult and tedious. The goal of research in this area is to help Veterans with the process of filing for claims. This projects specific goal is to evaluate the quality of embeddings of the datascraped text.

**Description of Dataset:**

Data was collected from this website: [www.knowva.ebenefits.va.gov](http://www.knowva.ebenefits.va.gov), a publicly accessible government website about the rules and regulations behind filing for claims with the Veterans. In the website, theres thousands of article webpages that contain these rules. We created our dataset by extracting the text from each of these pages. Each of these articles is also associated with a broader category.

**Preprocessing:**

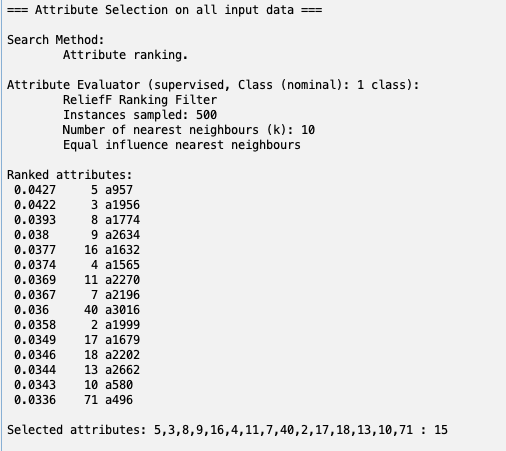
The text from these pages was broken into appropriately sized chunks, if the text in an article was over 500 words. Then, each chunk of text was fed through an openAI embedding model to create embeddings for each chunk of text. Duplicate texts were removed, and special non-recognized characters were also deleted.

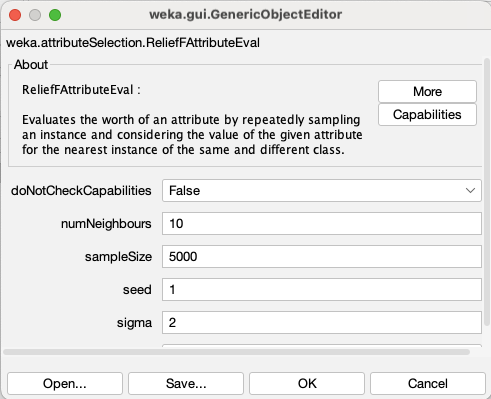
****

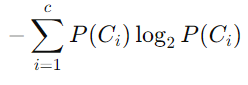
**Attribute Selection Algorithms:**

1. **(Non-Weka) No Attribute Removal**
   1. As a baseline method, we keep all the attributes in our data. This is also helpful as a control case to help us put into perspective the other results of our trials.
2. **PrincipalComponents** 
   1. PCA (Principal Component Analysis) is a attribute reduction technique that transforms the original attributes into a linear combination of attributes called Principal Components that tries to maximize the variance in the data. The PCA ranks n (# of original attributes) principal components by the amount of variance they capture. By keeping a variance threshold, we can remove some attributes which do not contain enough variance for our liking.
3. **ReliefFAttributeEval** 
   1. ReliefF attribute evaluation is an attribute selection algorithm that ranks the attributes by useful they are in distinguishing between the class types.

ReliefF is effective for handling noisy data since it focuses on the nearest neighbors, which is helpful for our data.

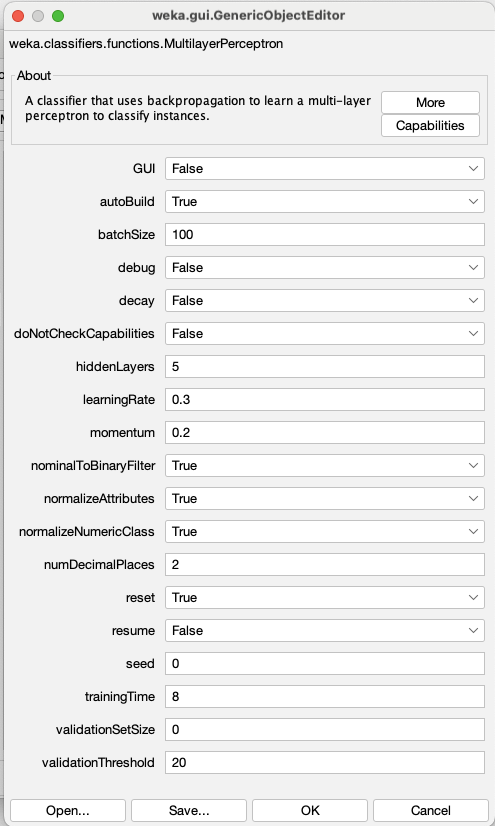
ReliefF attribute evaluation finds the similarities between different instances and the give a weight for each attribute in relation to the class .Here is an example of an attribute ranking using ReliefFAttributeEval:

Here are the hyperparameters we are initializing the algorithm with: 

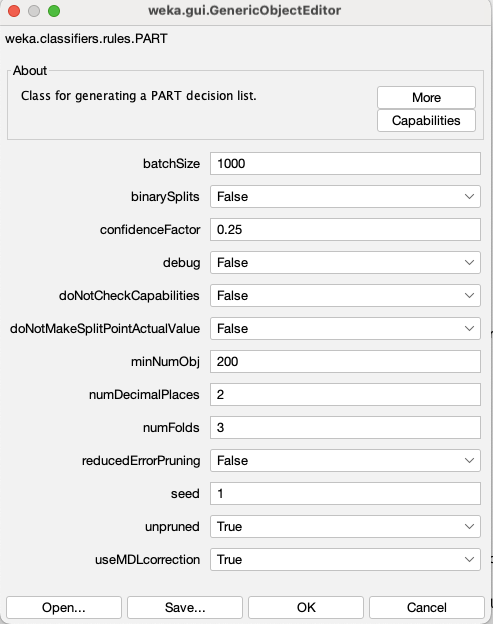
1. **GainRatioAttributeEval** 
   1. GainRatioAttribute Eval ranks attributes by their gain ratio in relation to the class attribute. Entropy is calculated by the following formula where P(Ci) is the probability of the attribute predicting the class. 
2. **Correlation**
   1. Correlation ranks attributes based on their correlation to each potential output class, and the attributes that have the highest correlation to one or more output classes are chosen

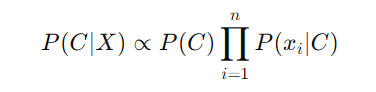
**Classification Algorithms:**

1. **Multi-layer Perceptron (MLP):**
   1. A Multi-layer Perceptron is a simplistic neural network that takes the instances as the input and the class as the output. We can tune hyperparameters for the MLP. Here are our hyperparameters:

****

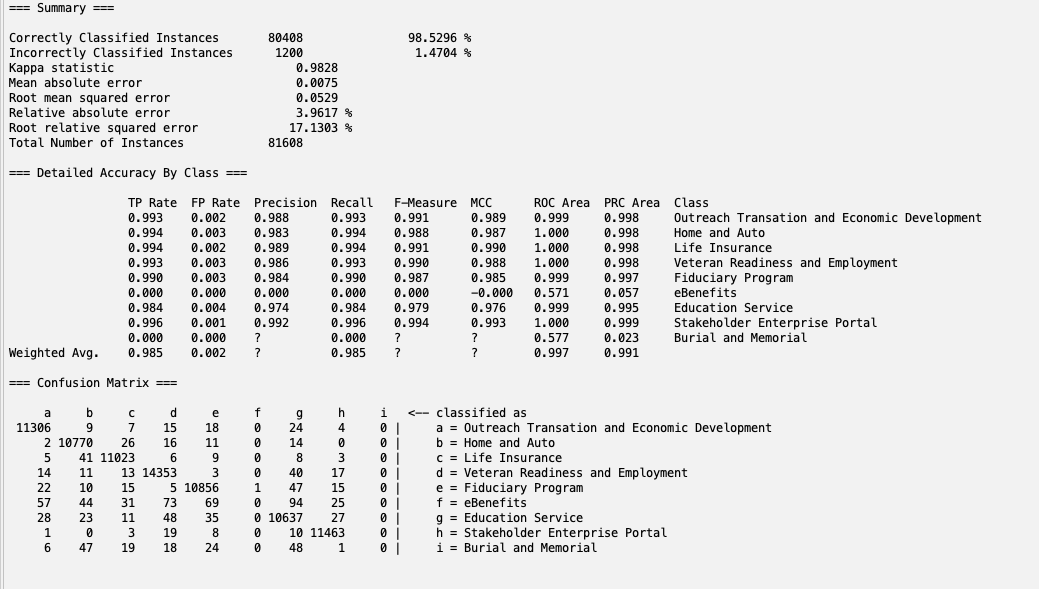
1. **PART (Partial Decision Trees)**
   1. PART (Partial Decision Trees) is a rule-based classification algorithm with a connection with the a decision tree. It works by making a decision tree for the dataset and then extracting rules from the trees and modifying the new trees until PART creates a set of rules for the dataset. Here are the hyperparameters for our PART initialization.

****

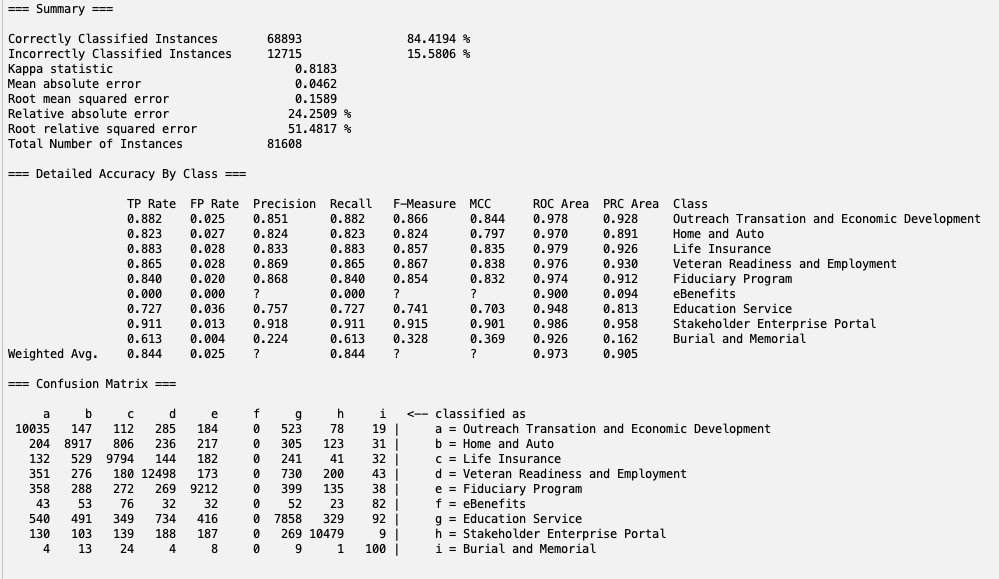
1. **Naive Bayes**
   1. Probabilistic Classifier based on calculating the prior probability based on the formula to the side. P(C|X) describes the function for the previous feature X for the class C. 
2. **J48**
   1. J48 is Weka’s implementation of the C4.5 algorithm, which works by building a decision tree by recursively splitting a dataset based on which feature has the most significant information gain.

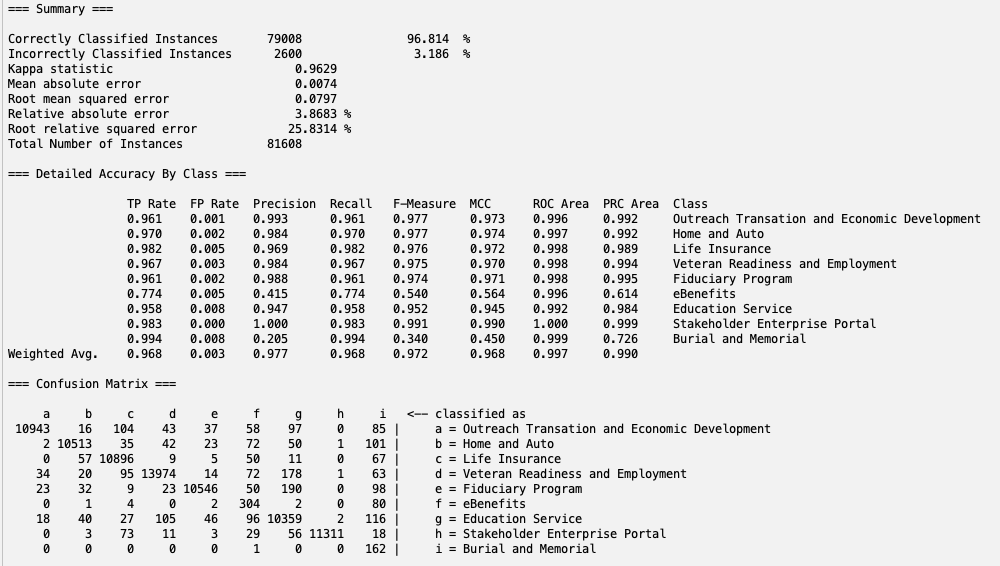
# Results:

No Attribute Removal with MLP:

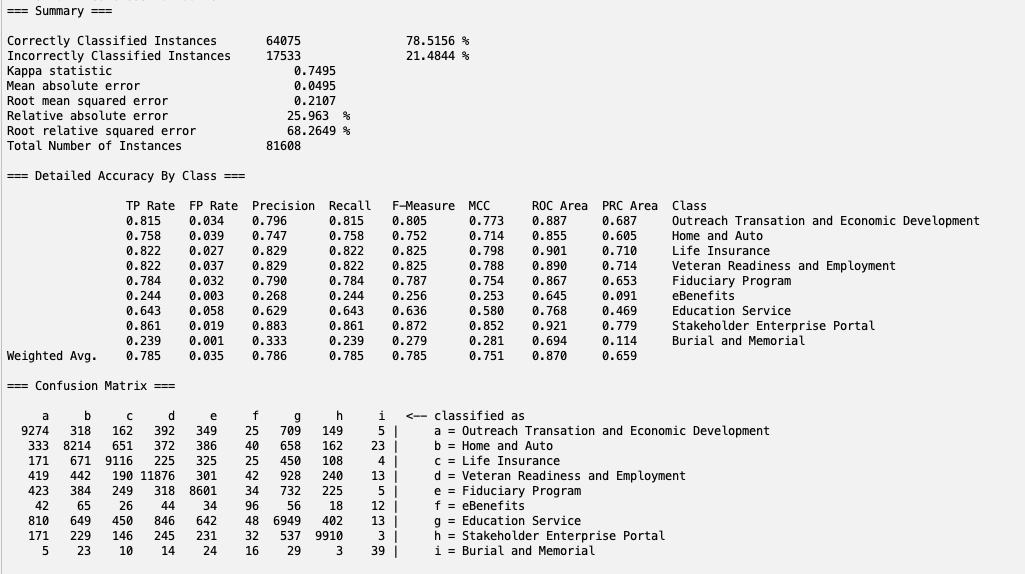


No Attribute Selection with PART:

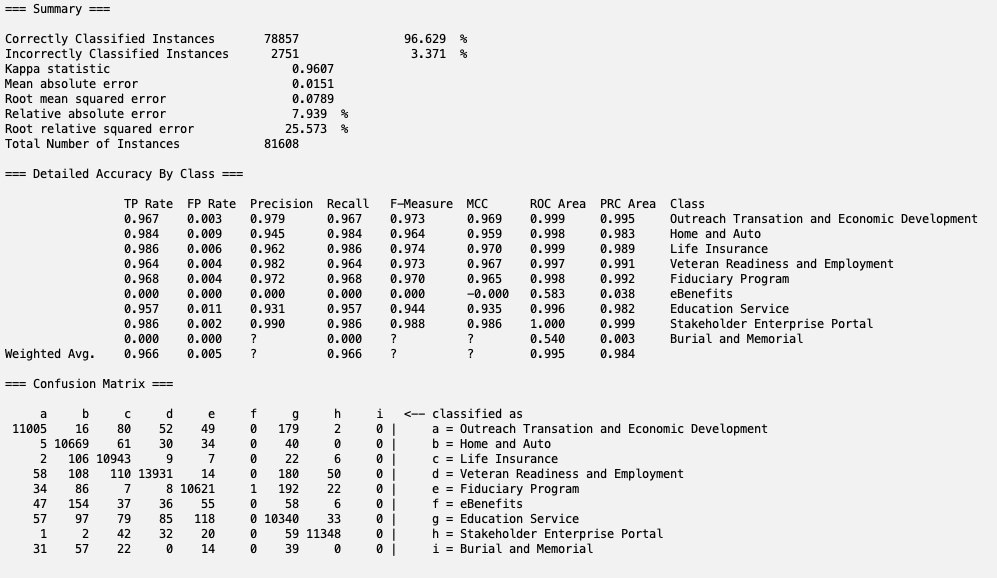


No Attribute Selection with Naive Bayes:  


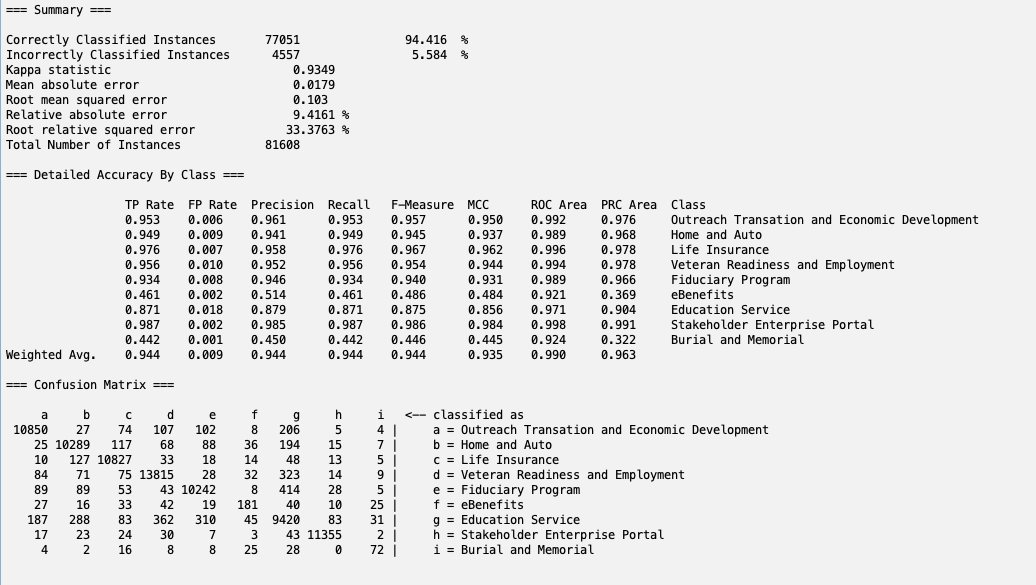
No Attribute Selection with J48:



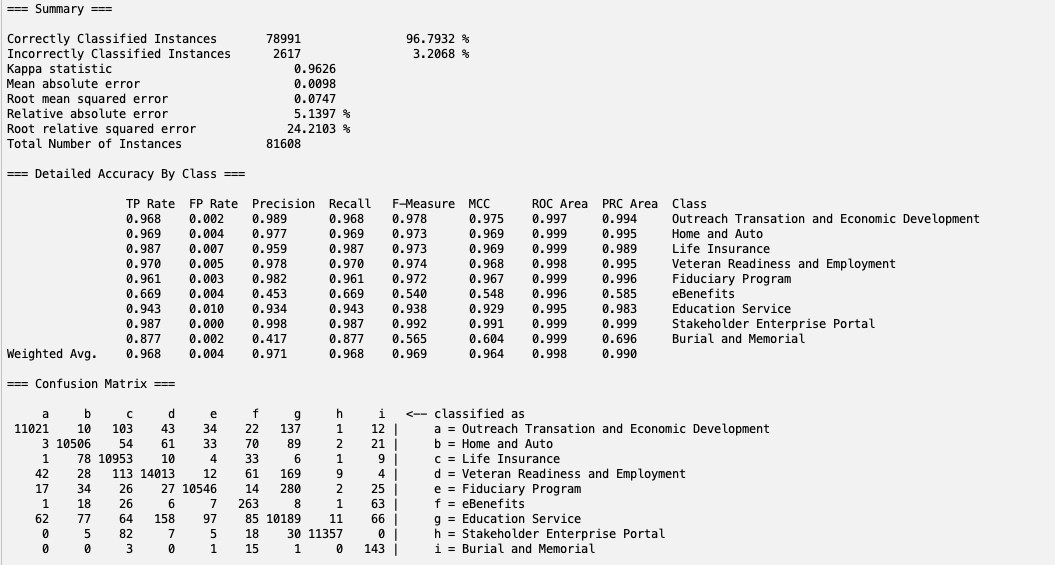
PCA with MLP:



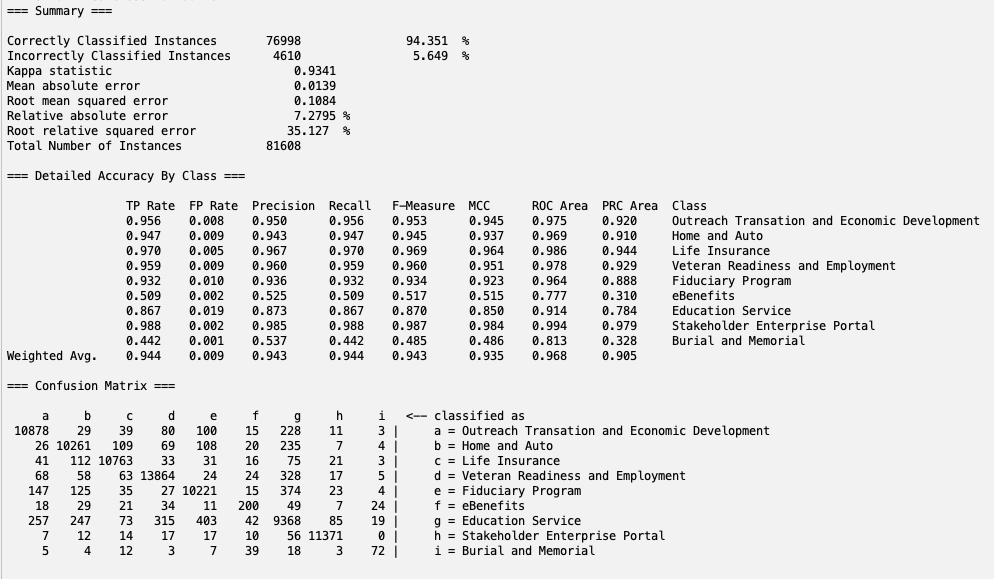
PCA with PART:



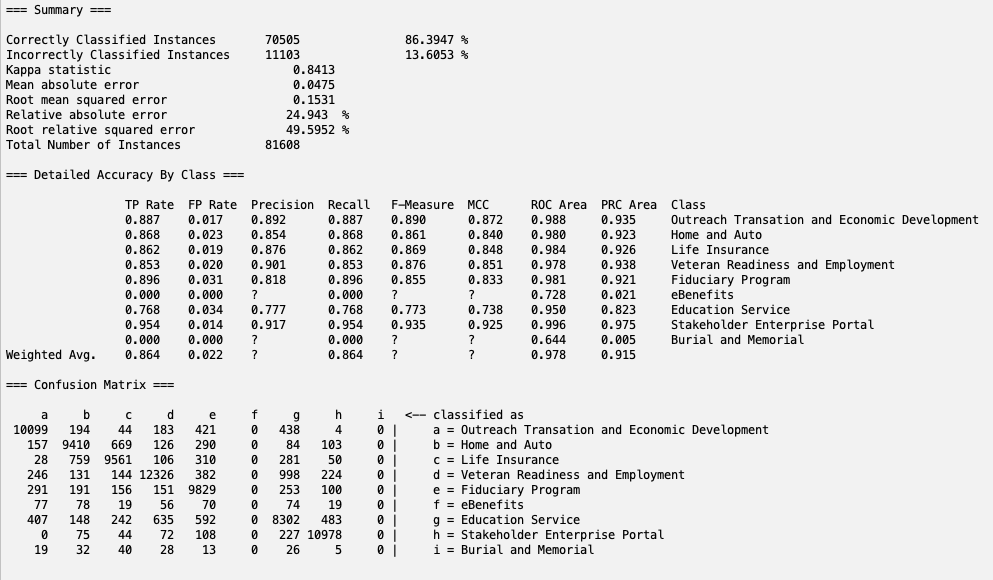
PCA with Naive Bayes:



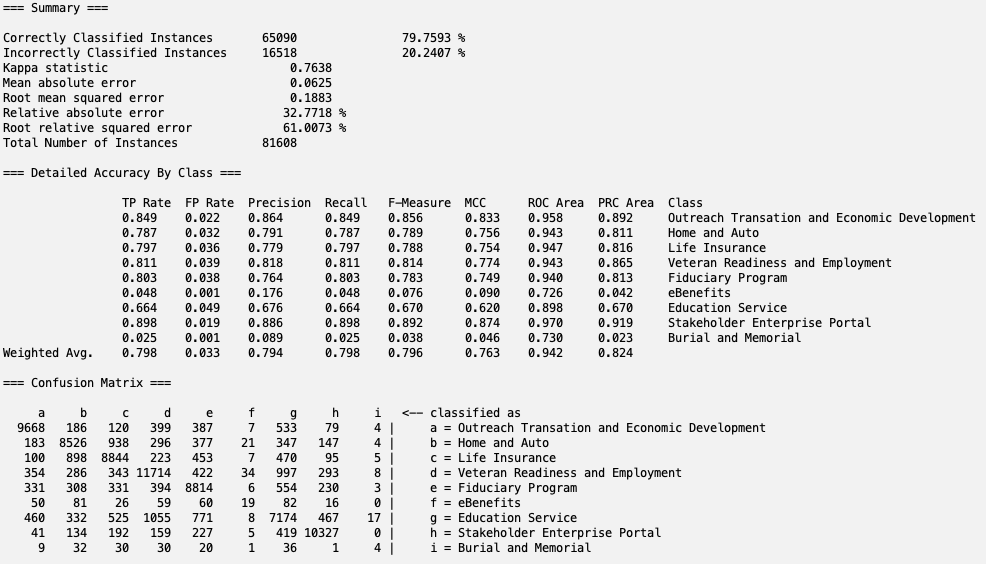
PCA with J48:



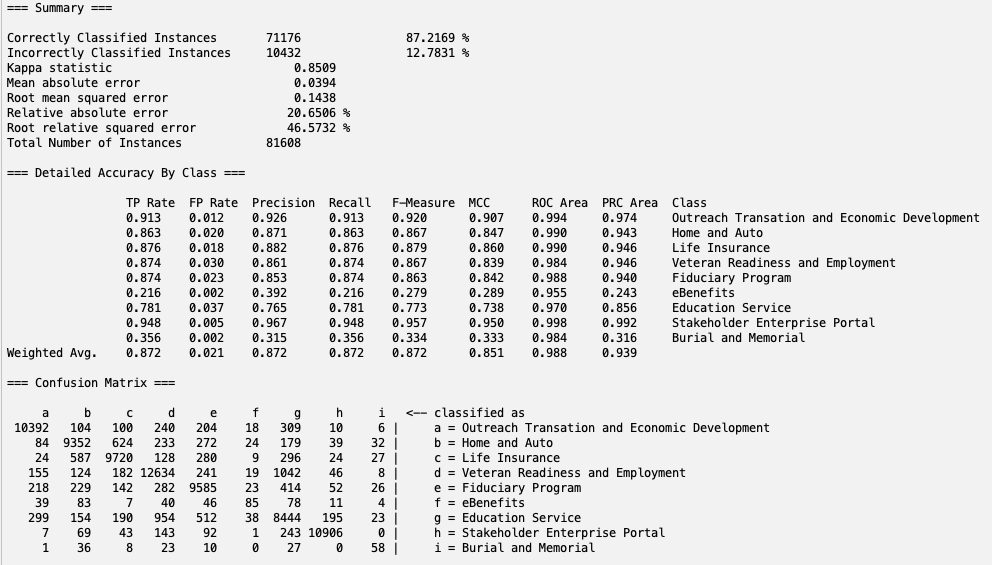
ReliefF with MLP:



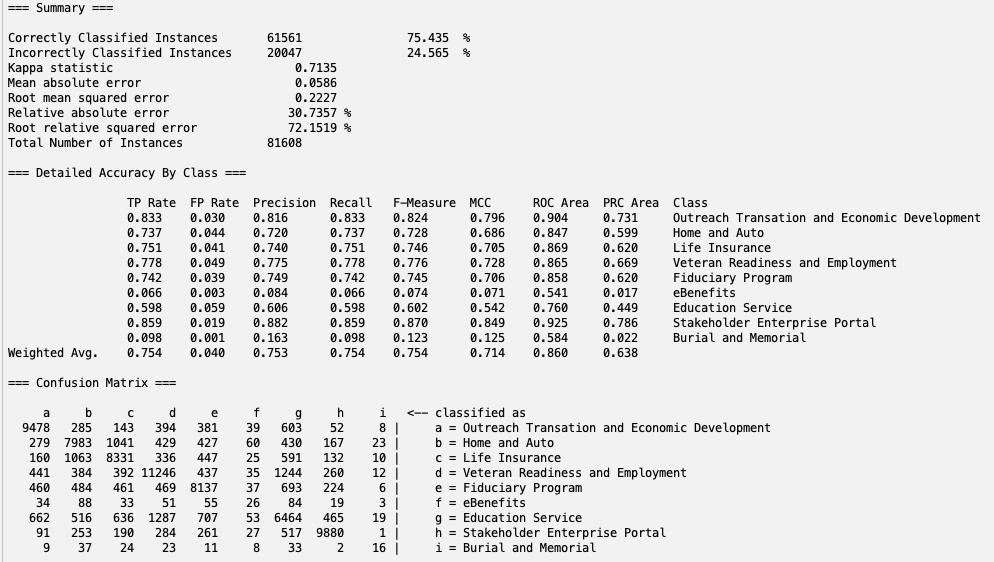
ReliefF with PART:



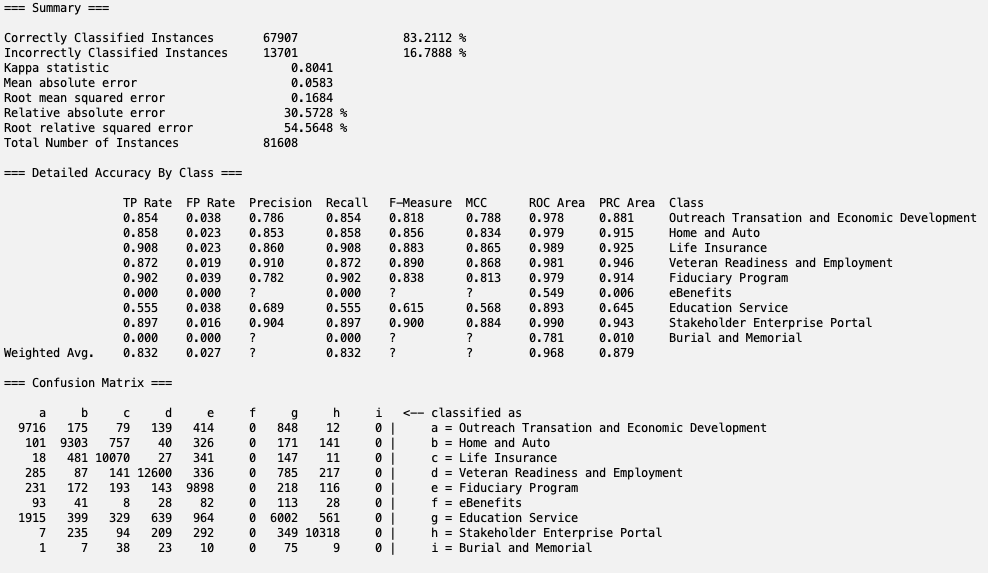
ReliefF with Naive Bayes:



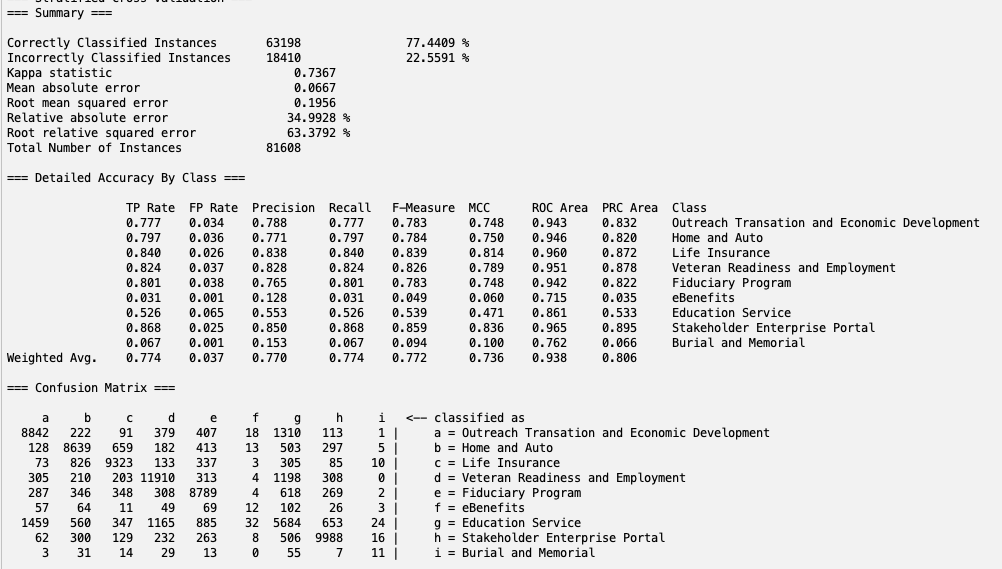
ReliefF with J48:



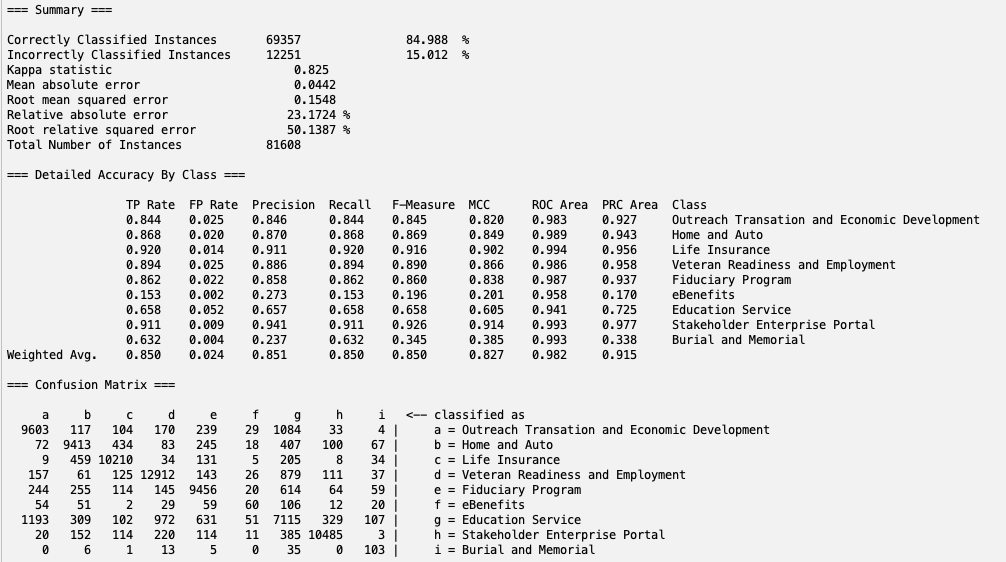
GainRatioAttributeEval with MLP:



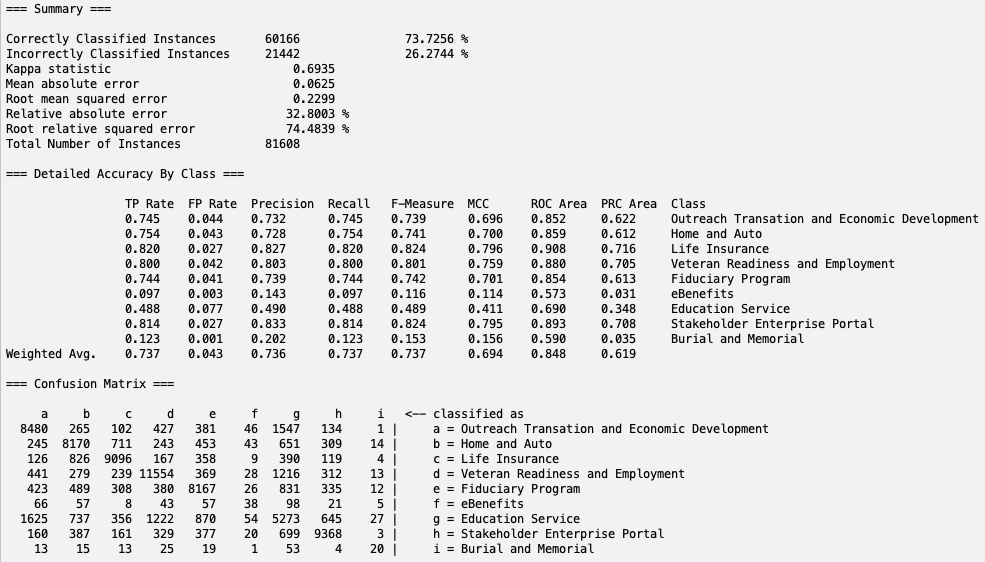
GainRatioAttributeEval with PART:



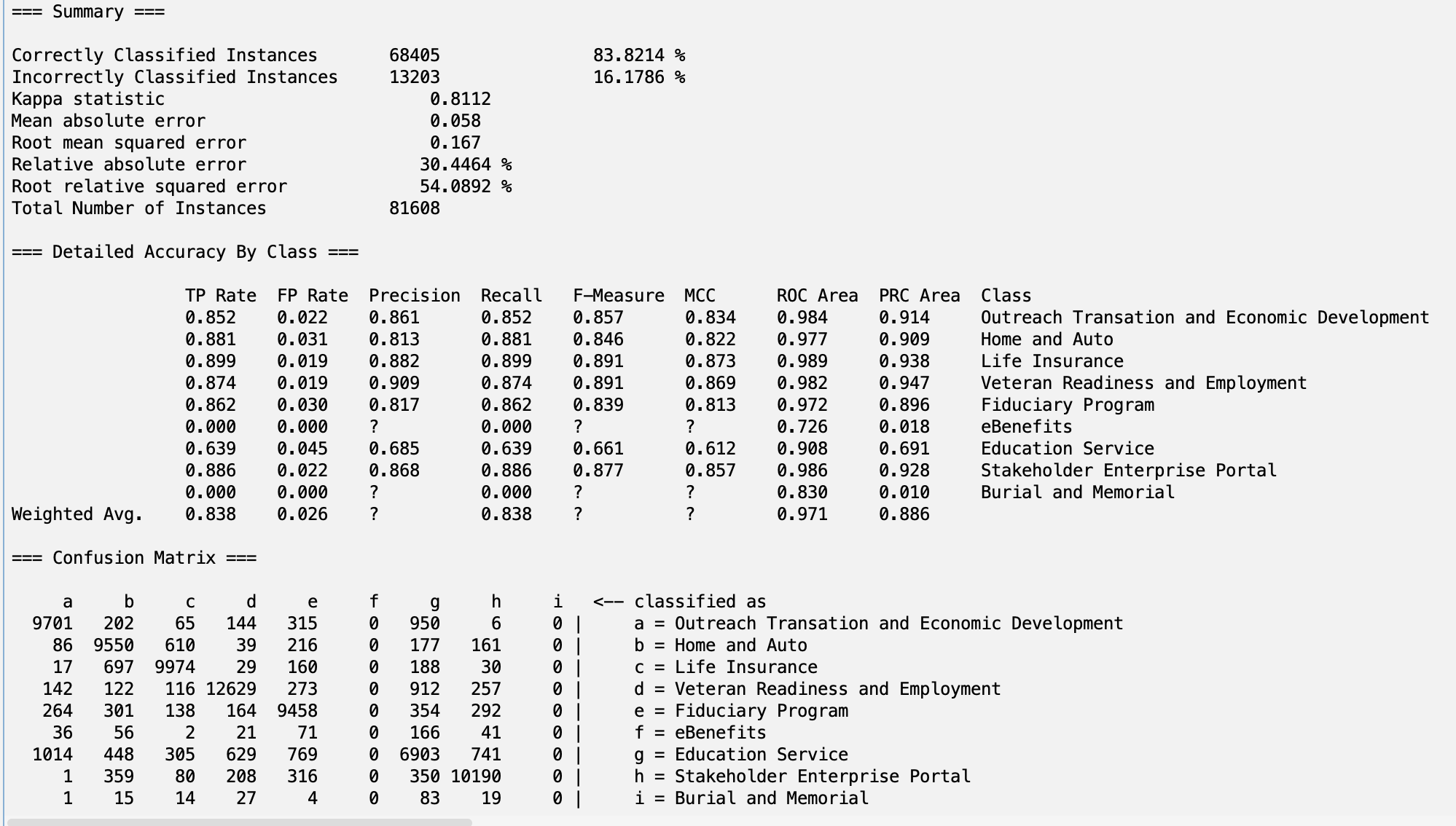
GainRatioAttributeEval with Naive Bayes:



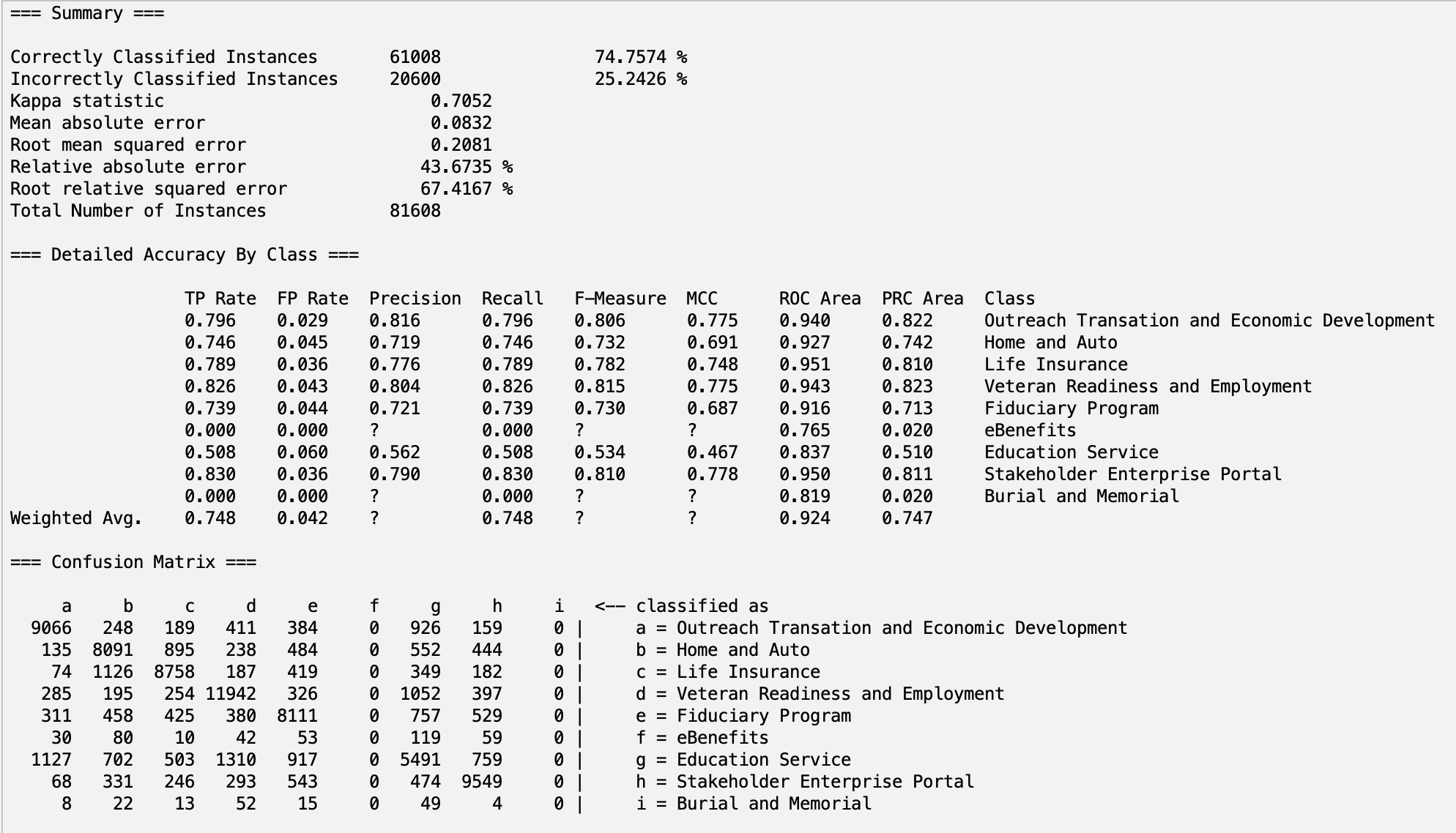
GainRatioAttributeEval with J48:



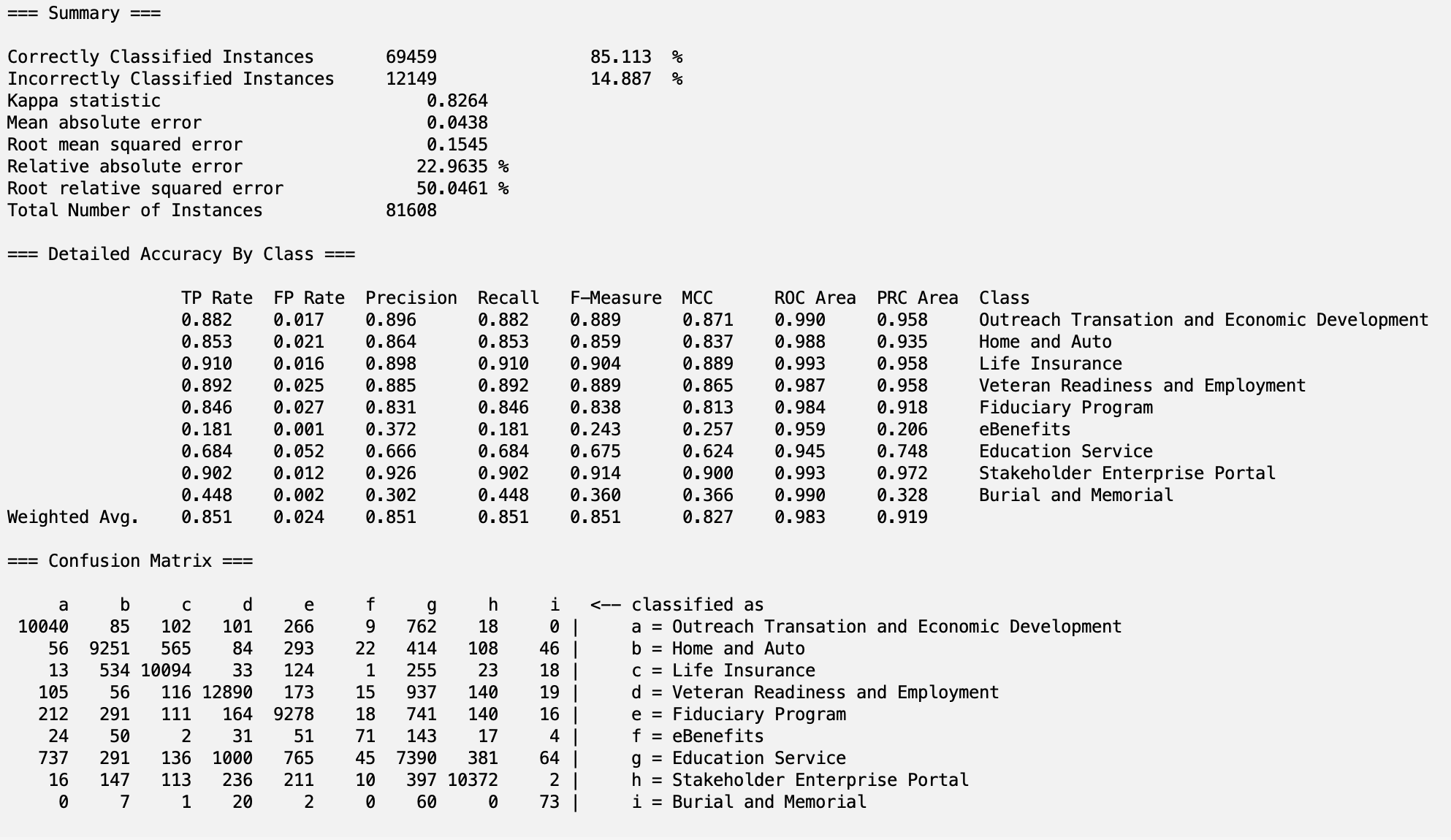
Correlation with MLP:



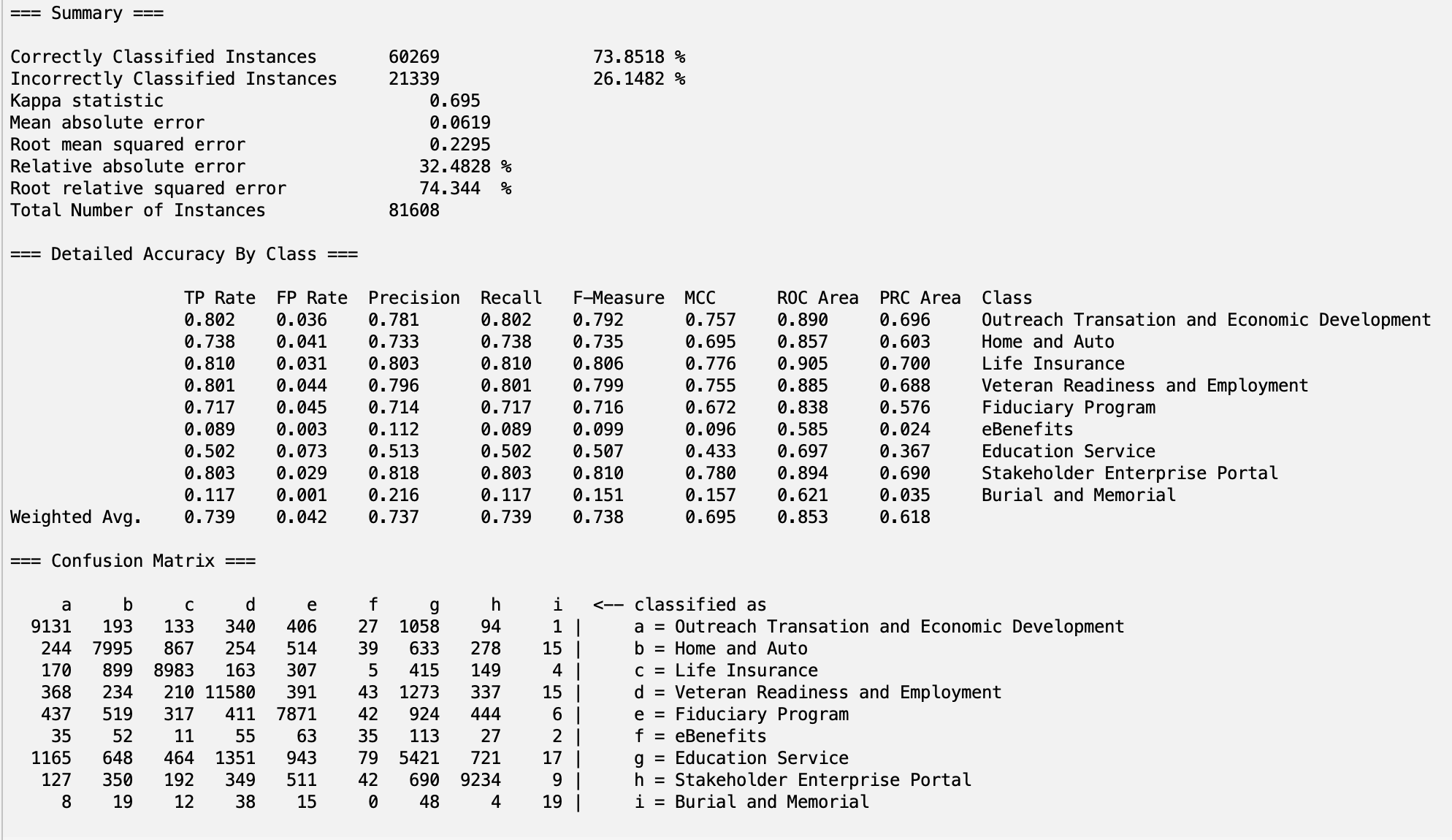
Correlation with PART:



Correlation with Naive Bayes:



Correlation with J48:



# Analysis:

| Table of Accuracy Values for Each Attribute Selection and Classification Combination | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Attribute (Right)  --------------------  Classification (Below) | No Attribute Removal | Principal Components Analysis | ReliefFAttributeEval | GainRatioAttributeEval | Correlation | Averages |
| Multilayer Perceptron | 98.53% | 96.63% | 86.39% | 83.21% | 83.82% | 89.72% |
| PART | 84.42% | 94.42% | 79.76% | 77.44% | 74.76% | 82.16% |
| Naives Bayes | 96.81% | 96.79% | 87.22% | 84.99% | 85.11% | 90.18% |
| J48 | 78.52% | 94.35% | 75.44% | 73.73% | 73.85% | 79.18% |
| Averages: | 89.57% | 95.55% | 82.20% | 79.84% | 79.39% |  |

After running 20 different classification models with 5 different Attribute Selection Algorithms combined with 4 different classification algorithms.

The best combinations were:

1. No Attribute Removal + Multilayer Perceptron (98.53%)
2. No Attribute Removal + Naives Bayes
3. Principal Component + Naives Bayes
4. Principal Component + Multilayer Perceptron

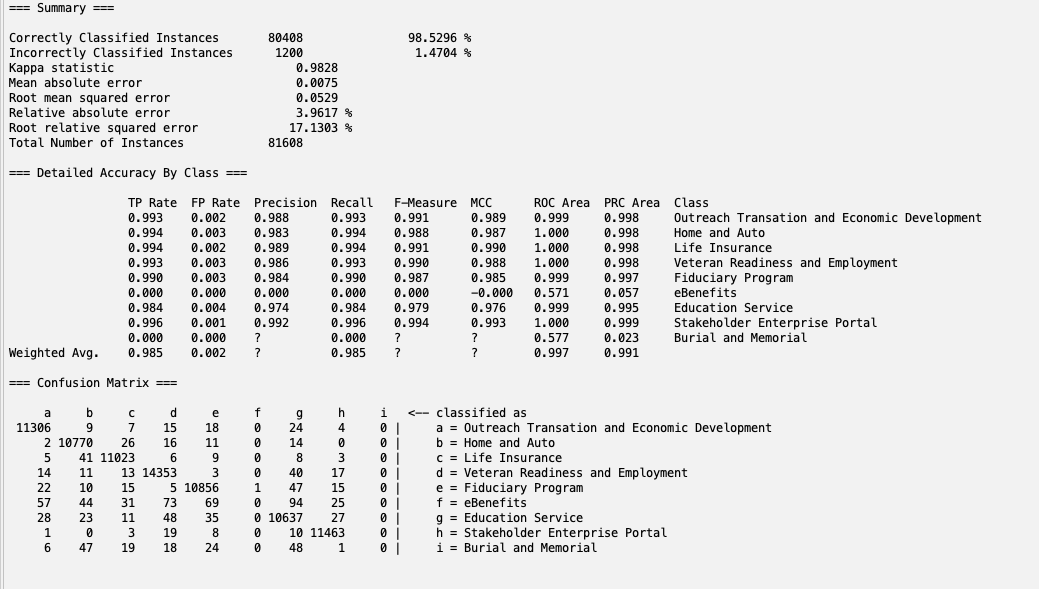
As they all achieved above a 95% accuracy.

The worst combinations were:

1. GainRatioAttributeEval + J48 (73.73%)
2. ReliefFAttributeEval + J48
3. GainRatioAttributeEval + PART
4. No Attribute Removal + J48
5. ReliefFAttributeEval + PART

After taking averages of accuracies for different attributes selections algorithms, **Principal Component Analysis** is the best for any classification task since the average is 95.55% across the four different classification methods.

Best Confusion Matrix:



If we look at all of the metrics for this classification, we can see that a majority of them are around 0.999, except the eBenefits class. We think this is because the eBenefits attribute has the least amount of instances. This would lead to the accuracy and confusion matrix for eBenefits to be much lower than other classes.

# Conclusion

Based on our results, we can conclude that **No Attribute Removal + Multilayer Perceptron** had the highest accuracy, 98.53%.

This is an interesting result as this tells us removing attributes, even the ones with less correlation or miniscule variance, still has an effect on the classification accuracy. However, running the model on the data with no attributes removed took 5 times longer, which would make sense as the dataset is much longer. If attribute number must be lowered, future projects with this dataset should utilize a Principal Component Analysis to transform data because it resulted in the highest average of classification accuracies compared to other attribute selection methods.

### How to reproduce our Optimal Model (No Attribute Selection + Multilayer Perceptron):

1. Open Weka and load the “embeddings\_82250\_attributes\_v2\_top100.csv” file from our project folder.
2. Click Classify in the top menu of Weka
3. Click Choose
4. Under the ‘Bayes’ folder, choose the **MultiLayerPerceptron** option
5. Click the word “**MultiLayerPerceptron**” to open the hyperparameter options
6. Choose **False** for GUI.
7. Choose **True** for autoBuild.
8. Set **100** for batchSize.
9. Choose **False** for debug.
10. Choose **False** for decay.
11. Choose **False** for doNotCheckCapabilities.
12. Set **5** for hiddenLayers.
13. Set **0.3** for learningRate.
14. Set **0.2** for momentum.
15. Choose **True** for nominalToBinaryFilter.
16. Choose **True** for normalizeAttributes.
17. Choose **True** for normalizeNumericClass.
18. Set **2** for numDecimalPlaces.
19. Choose **True** for reset.
20. Choose **False** for resume.
21. Set **0** for seed.
22. Set **8** for trainingTime.
23. Set **0** for validationSetSize.
24. Set **20** for validationThreshold.
25. Click nom (class) in the class selection area on the left middle of the page
26. Click Run
27. Wait for results

# Team Members and Tasks Performed:

Anurag Perakalapudi:

* Selected and Described Attribute Selection Algorithms
* Selected and Described Classification Algorithms
* Analyzed Results
* Wrote Conclusion
* Wrote Reproduction Procedures
* Created Presentation

Aneesh Kalla:

* Created webscraper and scraped text from VA website
* Preprocessed text
* Created embeddings
* Ran WEKA models
* Wrote introduction materials

# Appendix and Sources

Veterans Affairs Webpage:

Files attached with the report:

# References

[Weka class description (GainRatioAttributeEval)](https://weka.sourceforge.io/doc.dev/weka/attributeSelection/GainRatioAttributeEval.html#:~:text=Evaluates%20the%20worth%20of%20an,values%20as%20a%20separate%20value.)

[www.knowva.ebenefits.va.gov](http://www.knowva.ebenefits.va.gov)