**Jeffrey Tsai**

**Ling 406**

**Jeffrey5**

**4/1/2020**

**Assignment 3 – Using Scikit-Learn/Weka**

**Naives Bayes**

**Results:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Features Removed** | **None** | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** | **A7** |
| **Accuracy** | **79.20** | **80.26** | **79.52** | **79.14** | **19.12** | **79.82** | **79.26** | **79.18** |
| **Precision** | **67.90** | **69.33** | **76.01** | **75.72** | **13.70** | **78.84** | **76.61** | **76.04** |
| **F-measure** | **65.79** | **67.03** | **65.30** | **64.50** | **6.15** | **64.66** | **65.82** | **65.69** |
| **Recall** | **67.38** | **68.51** | **76.83** | **75.92** | **19.20** | **78.44** | **77.53** | **76.77** |

**Performance Table (performance with all the features - performance with all the features minus the current feature)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Features Removed** | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** | **A7** |
| **Accuracy** | **-1.06** | **-0.32** | **0.06** | **60.08** | **-0.62** | **-0.06** | **0.02** |
| **Precision** | **-1.43** | **-8.11** | **-7.82** | **54.2** | **-10.94** | **-8.71** | **-8.14** |
| **F-measure** | **-1.24** | **0.49** | **1.29** | **59.64** | **1.13** | **-0.03** | **0.1** |
| **Recall** | **-1.13** | **-9.45** | **-8.54** | **48.18** | **-11.06** | **-10.15** | **-9.39** |

**Decision Tree Classifier**

**Results:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Features Removed** | **None** | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** | **A7** |
| **Accuracy** | **94.88** | **95.00** | **95.10** | **94.08** | **32.02** | **94.38** | **95.46** | **95.14** |
| **Precision** | **86.00** | **84.16** | **89.25** | **88.19** | **20.16** | **87.81** | **90.25** | **88.67** |
| **F-measure** | **85.55** | **83.51** | **86.47** | **84.69** | **18.04** | **84.76** | **86.88** | **85.29** |
| **Recall** | **87.16** | **85.17** | **90.82** | **90.00** | **19.82** | **88.78** | **92.52** | **91.06** |

**Performance Table (performance with all the features - performance with all the features minus the current feature)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Features Removed** | **A1** | **A2** | **A3** | **A4** | **A5** | **A6** | **A7** |
| **Accuracy** | **-0.12** | **-0.22** | **0.8** | **62.86** | **0.5** | **-0.58** | **-0.26** |
| **Precision** | **1.84** | **-3.25** | **-2.19** | **65.84** | **-1.81** | **-4.25** | **-2.67** |
| **F-measure** | **2.04** | **-0.92** | **0.86** | **67.51** | **0.79** | **-1.33** | **0.26** |
| **Recall** | **1.99** | **-3.66** | **-2.84** | **67.34** | **-1.62** | **-5.36** | **-3.9** |

**Notes:**

I originally used Weka because it is a newer way of allowing us to understand how machine learning can be applied and used in a newer, simpler way. It was clearly obvious after using it, however, that there is still a lot of room for improvement. Carrying out this task in Python through NLTK or through Scikit Learn should yield different if not better results. Utilizing a tool like Weka over more traditional code for this task is daunting but feels comparable to choosing to code a task out in Java over x86; it seems to save you lots of time and effort, but it doesn’t give you full control of everything inside. It’s hard to pinpoint errors and bugs.

In the end, I used Scikit-learn and Python to calculate my performance tables. I tried many different pre-processing methods as suggested by my peers and classmates on Piazza, such as the DictVectorizer or the OneHotEncoder. The most effective method was to utilize the LabelEncoder to uniquely identify the words and classes in a unique part of speech.

**Conclusion:**

1. **Which is the best machine learning model (classifier) for this task? You need to discuss this per metric used to compute the performance.**

Based off of the data I have gathered, the best classifier for this specific task seems to be the decision tree classifier. With our results, we can see that the decision tree classifier is a better classifier to use than the Naïve Bayes by almost 20%. This is true when the data has already been pre-processed. I would further predict that this difference would be even bigger if the data isn’t pre-processed.

1. **Which features contributed the most to the performance (precision, recall, accuracy, and F-measure)? Which contributed the least? (you need to do this for each machine learning model considered here)**

The fourth feature seems to contribute the most to performance for both the Decision Tree and the Naive Bayes classifiers. For the Decision Tree Classifier, it seems that the other attributes contribute almost nothing to the performance. For Naive Bayes, however, the features at the very edge, furthest away from a4, seem to contribute the least to performance. The accuracy increases if a1, a2, a6, and a7 are removed, and the further the distance the feature removed has with a4, the more accurate it becomes. This is most likely because a1-a3 and a5-a7 provides context for a4.

1. **How good is this feature set (i.e., the seven attributes) for this task? (based on the answer at Question 2)**

It feels like the feature set could be improved. Attribute four is definitely the attribute of interest as it seems to be the attribute that contributes most to performance. The other attributes seem to contribute very little to the overall performance for both the Naïve Bayes and the Decision Tree Classifier. Though there are small differences, nothing seems to be marginal enough to be contributing to the overall performance. It could be worth it to only train the model with the fourth attribute by itself. This suggests that perhaps the feature set lacks context. The fact that removing certain attributes actually increases accuracy suggests the idea that adding some of these attributes is detrimental to our results. This suggest the possibility of too many attributes.

**If you had to solve the problem in a different (and probably more efficient way), which attributes would you choose? (write 1-2 short paragraphs about the features sets you might want to try for this problem).**

If we analyze our performance tables, we can see that the fourth attribute is the most important attribute of this task. It provides a marginal difference in accuracy and therefore it should be the center of attention for this task. Whether the other attributes should be chosen, however, is a different story.

Based off of our results, if we were to use the Decision Tree Classifier for this task, we should only consider the fourth attribute, since it seems that the other attributes contribute very little towards performance if not negatively affect it. If we were to use Naïve Bayes for this task, attributes 3 and 5 should also be chosen alongside attribute 4 because they seem to also contribute to performance. This would make sense since attributes 3 and 5 are right next to attribute 4.