Keshav Malhotra(NETID : kmahtr3)

Model 1: Decision Tree Classifier

Precision : 0.863
Recall : 0.711
F Measure : 0.710
Accuracy : 0.711

	a1	a2	a3	a4	a5	a6	a7
Precision	0	0	0	0.564	0	0	0.0010
Recall	0	0	0	0.439	0	0	0
Accuracy	0	0	0	0.438	0	0	0.0004
F Measure	0	0	0	0.470	0	0	0

Table 1 : Feature Contribution for Decision Tree Classifier

Model 2: Naive Bayes Classifier

Precision : 0.637
Recall : 0.553
F Measure : 0.506
Accuracy : 0.5528

	a1	a2	a3	a4	a5	a6	a7
Precision	0.0014	0.0200	0.0320	0.2940	0.0460	0.0130	0.0010
Recall	0.0210	0.0129	0.0240	0.2220	0.0250	-0.0090	-0.0209
Accuracy	0.0212	0.0034	0.0242	0.2222	0.0249	-0.0090	-0.0208
F Measure	0.0260	0.0140	0.0310	0.2420	0.0280	-0.0060	-0.0240

Table 2 : Feature Contribution for Naive Bayes Classifier

Q1
The best machine learning model for this task is the Decision Tree Classifier since it's accuracy(with all its features) is 71.1% compared to 55.28% for the Naive Bayes Classifier. It has also significantly better Recall, F Measure as well as Precision. This is primarily because the Naive Bayes classifier used for this dataset isn't effective as it doesn't have any information about the preceding or following word. For instance, there's no information about the Part of speech for the previous and the following words. Just using the surrounding words as features is not an effective way of determining part of speech also since Naive Bayes assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Decision Tree Classifier, on the other hand, however, just primarily focuses on the simplest possible way of classifying the word in question. The prediction of the decision tree is based on the most common POS tag for the given word, and isn't really affected by surrounding data.

Q2
Feature a4 contributed most to the performance of Decision Tree classifier. Other features had less significant or no impact at all. After removing all features but a4, the accuracy of the Decision Tree Classifier stayed almost the same (71.04%).

Feature a3, a4 and a5 contributed most to the performance of Naive Bayes, with a4's contribution being the highest, since that's the word being tagged.

Q3
The given feature set is not significantly useful for this POS tagging task. Removing any feature except a4(the word that's being tagged) doesn't have a significant impact for the Decision Tree Classifier. For Naive Bayes, removing all features but a4 actually improves the accuracy by approximately 8% which means that the feature set provided with the given data actually hurts the Naive Bayes Classifier instead of providing additional tools to improve its performance.

Q4
To get more accuracy, I would use Naive Bayes classifier but instead of using word values as surrounding context for POS tagging of a word, I would use the POS tags of the surrounding words since that would add more contextual information which would improve the accuracy of the model. I would also restrict the features to just 3, so one word before and one word after the given word that is being tagged, should be sufficient amount of context as going further away from the given word would have very less impact on the classification. The dataset will be smaller and it would reduce the program's running time.

I would also try tuning my classifier by tweaking it’s input parameters. Using a larger dataset for training can help improve the accuracy. Working in log space might be more efficient since the probabilities calculated for large number of cases for POS tags would be very low and multiplying small numbers would result into even smaller numbers. Using log of probabilities can avoid the problem of under runs.