Keaton Spiller

CS 445 Winter 2022

Programming Assignment #2

Classification with Gaussian Naïve Bayes

In this programming assignment I strictly used pandas, NumPy, and math without the use of scikit. I struggled with every section of the algorithm, checking spam and non spam sequentially wasn't intuitive to me. It took me longer to figure out the right way to loop through each email. I originally tried to compare each feature to each email but this was the wrong approach. Once I stepped back it was easier to see we were comparing each email with every feature. I initially thought needed to add the prior with each N feature, and had 1 gaussian for both classes. However, the correct approach was to calculate both every time for a separate Gaussian, mean, standard deviation, and probability.

I spent a large amount of time trying to figure out how to incorporate the argmax function. However, I already had the probabilities I needed to compare, and realized all I needed to do was a comparison. In the previous assignment we used argmax to find the index of the label that had the highest probability, however since I needed to check two classes instead of 10 the problem became simple.

	Predicted Spam	Predicted Non_Spam
Actual Spam	882.0	453.0
Actual Non_Spam	30.0	936.0

Accuracy: 0.79 Precision: 0.66 Recall: 0.97 Runtime: 8.3 seconds

I don't think the attributes are independent because I think the accuracy would be a lot higher if we were working with strict independent attributes. However, 79% accuracy is adequate for many image classification applications and is fast compared to neural network with a run time of 8.3 seconds. Naïve Bayes would do poorly with many non independent emails because it would break in cases where the gaussian's top half zeros out and where the results cancel out. This algorithm often ran into a log(0) in the top half of the gaussian, which I substituted for negative infinity to ignore the probability and treat it as insignificant. The precision being low at 66% showed flexibility in the predictions with a high ability to recall and since we don't want to memorize the data a lower precision could be a good thing.

Experimenting with a K value in the Gaussian using a variation of the Laplace smoothing techniques I was able to get the accuracy to 0.91 % precision to 0.92 % and recall to 0.86 %.

	Predicted Spam	Predicted Non_Spam
Actual Spam	780.0	70.0
Actual Non_Spam	132.0	1319.0