Artificial Intelligence

Group 3B
Rick Fontein s1483870
Niels Geuze s1478494
Mark van Kruistum s1460536
Kasper de Kruiff s1479733
Rens Kruining s1351524

Intro:
Our apologies for adding the data files to the project. We did this to make sure the code would be executable since it has to be provided in a certain manner. The GUI for part D was made using the form builder of IntelliJ. Therefore, we included a .jar file. When ran from the command line, the console will show results from part A and open a GUI for part D. Full source code is available on https://github.com/Telluur/MachineLearning

A:

The results of the male/female blogs is with smoothing k=1 66% and with k=0.00001 74% which was the best we found, with the ham/spam emails with k=1 the accuracy is 98.97% which was also the highest we got if you go either lower or higher it won't change or will lower, with k=0.001 the accuracy is 98.63%. The script that gives these results is MachineLearning.java in package A, you can run it without doing anything and on the top you can see 2 variables that you can change to adjust the smoothing (BLOG_SMOOTHING and EMAIL_SMOOTHING).

B:

NBC:

Male/Female train/test:

```
 a  b   <-- classified as
18  7 |  a = F
10 15 |  b = M
```

Spam/Ham 10- folds cross-validation

```
   a    b   <-- classified as
2366   46 |    a = ham
  21  460 |    b = spam
```

Accuracy = 0.66

Precision = 0.662

Recall = 0.66

F-measure = 0.659

Accuracy = 0.977

Precision = 0.978

Recall = 0.977

F-measure = 0.977

C

J48:

Male/Female train/test:

```
 a  b   <-- classified as
15 10 |  a = F
 8 17 |  b = M
```

Accuracy = 0.64

Precision = 0.641

recall = 0.64

F-Measure = 0.639

Spam/Ham 10- folds cross-validation

```
a    b   <-- classified as
2358   54 |    a = ham
64  417 |    b = spam
```

Accuracy = 0.959

Precision = 0.959

recall = 0.959

F-Measure = 0.959

BayesianLogisticRegression:

Male/Female train/test:

```
 a  b   <-- classified as
18  7 |  a = F
11 14 |  b = M
```

Accuracy = 0.64

Precision = 0.644

recall = 0.64

F-Measure = 0.638

Spam/Ham 10- folds cross-validation

```
a    b   <-- classified as
2407    5 |    a = ham
  11  470 |    b = spam
```

Accuracy = 0.994

Precision = 0.994

recall = 0.994

F-Measure = 0.994

As you can see for the Male/Female data the NBC is the best classifier although it is close, this is because it is very hard to say if an email is male or female this gives a lot of variation and logistic regression is not very good when that happens. Decision trees are in general not ideal for text which you can see in the results as well since it is the worst in all cases. This is why for determining if a certain text is written by a Male or Female NBC is the best solution out of these three. Spam and Ham is a lot easier to separate, the line between them is a lot clearer which makes it so that Logistic regression becomes really good and you can see that in the result as it is nearly perfect, while the others still have errors >2% of the time.

D:

We made a GUI that works with the classifier made in part A. When we run the program, we get a prompt to select either the blogs or the emails. We then train as described in part A. A new window opens with two text fields and three buttons.

Fields: A big field to show the message and an information display. The information display gives us both a static and a learning prediction. The static prediction is solely based on the train data, the learning prediction is based both on the train data as well as the previous shown messages. Since we also know the actual classification of the message we are able to display this as well.

Buttons: Set 1 (Ham or Male), Set 2 (Spam or Female) and a 'show interactive learner' button. When the set buttons are pressed, the message on screen will be added to that particular set. The show interactive learner button will keep on pressing the set1/set2 button based on actual type of the message till it finds a message where the learning and the static prediction differ. We made some screenshots of this below.

Please note that to make this work (and thus show that it actually learns) we had to programmatically move data from the train sets to the test sets. So the results will differ from those of part A.