Stylometry in R

Dr Gordon Ross

Source Code Files

The Learn page has a file called stylometryfunctions.R which contains some functions for applying discriminant analysis and KNN to a literary corpus. You can use these in your assignments

To load the functions, save the file into a directory and use:

```
source("~/directoryname/stylometryfunctions.R")
```

where ‘directoryname’ is the location on your computer where you saved your file. The above is for Mac/Linux. On windows, the directory name will look something like:

```
C:\\Downloads\\stylometryfunctions.R
```

Alternatively you could simply copy/paste the contents of the stylometryfunctions.R into the R terminal, but this is bad practice.

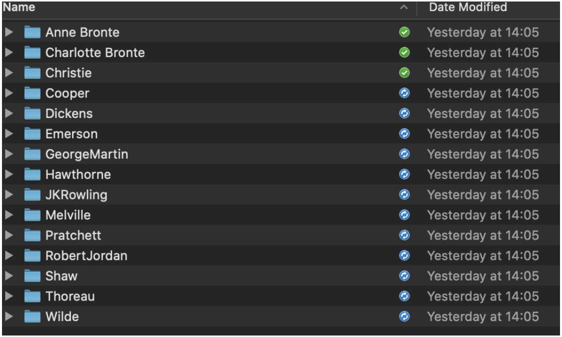
Data Input

In R, we can read data in from files using the read.csv() command. We used this in the lectures to read in the J.K Rowling files.

We will now do an extended example using a larger corpus. This will let us explore how performance can be analysed using cross validation

Data Input

This corpus contains the following authors.



Name		Date Modified
Anne Bronte	✓	Yesterday at 14:05
Charlotte Bronte	✓	Yesterday at 14:05
Christie	✓	Yesterday at 14:05
Cooper	⟳	Yesterday at 14:05
Dickens	⟳	Yesterday at 14:05
Emerson	⟳	Yesterday at 14:05
GeorgeMartin	⟳	Yesterday at 14:05
Hawthorne	⟳	Yesterday at 14:05
JKRowling	⟳	Yesterday at 14:05
Melville	⟳	Yesterday at 14:05
Pratchett	⟳	Yesterday at 14:05
RobertJordan	⟳	Yesterday at 14:05
Shaw	⟳	Yesterday at 14:05
Thoreau	⟳	Yesterday at 14:05
Wilde	⟳	Yesterday at 14:05

Each folder contains the books written by one author. You will find this corpus in the ‘corpus.zip’ file that I will provide.

Data Input

You can load the corpus by using the loadCorpus() function from the stylometryfunctions file. The first argument is the directory of function words (this will depend on where you unzip the file)

```
M <- loadCorpus("~/Corpus/FunctionWords/",
  "frequentwords70")
```

M is a **list** which contains the corpus and various pieces of information. To see the authors in the corpus:

```
> M$authornames
 [1] "Anne Bronte"       "Charlotte Bronte" "Christie"
 [4] "Cooper"            "Dickens"          "Emerson"
 [7] "GeorgeMartin"      "Hawthorne"        "JKRowling"
[10] "Melville"          "Pratchett"        "RobertJordan"
[13] "Shaw"              "Thoreau"          "Wilde"
```

Data Input

The features are in the ‘features’ element of M:

```
> M$features[[1]]
       V1  V2 V3  V4   V5  V6  V7   V8  V9  V10 V11  V12 V13 V
[1,] 1283 304 14 196 2670  93  96  661 407  507 141  705 261
[2,] 2728 650 10 337 6489 237 375 1435 965 ...
```

This is a 2 row matrix containing the 71 features for the two books written by the first author (Anne Bronte, as in previous slide). Similarly:

```
M$features[[10]]
```

would give the features for the books written by the 10th author (Melville).

Training-Test Sets

To assess performance of the classifiers, we will create a test set which consists of a single book written by each of the 15 authors:

```
traindata <- M$features
testdata <- NULL
testlabels <- NULL #true authors for the test set

for (i in 1:length(traindata)) {
  #select a random book by this author by choosing a row (= book)
  testind <- sample(1:nrow(traindata[[i]]), 1)

  #add this book to the test set
  testdata <- rbind(testdata, traindata[[i]][testind,])
  testlabels <- c(testlabels, i)

  #now discard the book from the training set
  traindata[[i]] <- traindata[[i]][-testind,,drop=FALSE]
}
```

Training-Test Sets

To predict the author of the 15 test set books using Multinomial discriminant analysis, you can use the discrminantCorpus() function from the stylometryfunctions file:

```
preds <- discriminantCorpus(traindata, testdata)
sum(preds==testlabels)/length(testlabels)
[1] 0.9333333 #93\% accuracy
```

Since the training/test set split is random, you may get a slightly difference accuracy number to this.

Training-Test Sets

Similarly to predict the author of the 15 test set books using KNN, you can use the discrminantCorpus() function from the stylometryfunctions file:

```
> predsKNN <- KNNCorpus(traindata, testdata)
> sum(predsKNN==testlabels)/length(testlabels)
[1] 1 #KNN got 100\% accuracy here
```

Since the training/test set split is random, you may get a slightly difference accuracy number to this.

Cross-Validation

Next we will perform leave-one cross validation by looping through each author. For each author, we also loop through each of their books. We create a test set consisting only of that book, and a training set which contains all the corpus except that one book. The book is then classified.

We then repeat for each of the other books in the corpus.

Cross-Validation

```
predictions <- NULL
KNNpredictions <- NULL
truth <- NULL
features <- M$features
for (i in 1:length(features)) {
  for (j in 1:nrow(features[[i]])) {
    testdata <- matrix(features[[i]][j,],nrow=1)
    traindata <- features
    traindata[[i]] <- traindata[[i]][-j,,drop=FALSE]

    pred <- discriminantCorpus(traindata, testdata)
    predictions <- c(predictions, pred)

    pred <- KNNCorpus(traindata, testdata)
    KNNpredictions <- c(KNNpredictions, pred)
    truth <- c(truth, i)
  }
}
```

Cross-Validation

We can see the final accuracies:

```
> sum(predictions==truth)/length(truth)
[1] 0.9821429
> sum(KNNpredictions==truth)/length(truth)
[1] 1
```

Cross-Validation

```
> library(caret)
> confusionMatrix(as.factor(predictions), as.factor(truth))

          Reference
Prediction  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
        1   2  0  0  0  0  0  0  0  0  0  0  0  0  0  0
        2   0  2  0  0  0  0  0  0  0  0  0  0  0  0  0
        3   0  0 62  0  0  0  0  0  0  0  1  0  0  0  0
        4   0  0  0  3  0  0  0  0  0  0  0  0  0  0  0
        5   0  0  0  0 15  0  0  0  0  1  0  0  0  0  0
        6   0  0  0  0  0  2  0  0  0  0  0  0  0  0  0
        7   0  0  0  0  0  0  5  0  0  0  0  0  0  0  0
        8   0  0  0  0  0  0  0  2  0  0  0  0  0  0  0
        9   0  0  0  0  0  0  0  0  9  0  0  0  0  0  0
        10  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0
        11  0  0  0  0  0  0  0  0  0  0 41  0  0  0  0
        12  0  0  0  0  0  0  0  0  0  0  1 14  0  0  0
        13  0  0  0  0  0  0  0  0  0  0  0  0  3  0  0
        14  0  0  0  0  0  0  0  0  0  0  0  0  0  2  0
        15  0  0  0  0  0  0  0  0  0  0  0  0  0  0  2
```

Federalist Papers

Federalist Papers

Let’s apply some of the techniques we have learned to another real-world disputed authorship problem.

We will study the Federalist PApers, which we briefly discussed in the first lecture of this course.

Background

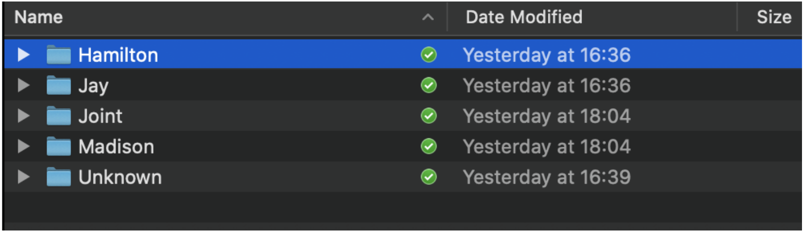
The Federalist Papers are a historically significant set of political essays written in 1787-1788 by three of the Founding Fathers of the United States of America (Alexander Hamilton, James Madison, and John Jay). They consist of 85 essays, where each was written by a single one of the above three authors

For 73 of these essays, the authorship is known and agreed on by historians. However the author of the remaining 12 essays is unknown, although it will be one of Hamilton, Madison, and Jay.

You can find the Federalist papers corpus in the federalist.zip file which I will make available.

Data

When you unzip this file, you will see the following file structure:



Name		Date Modified	Size
Hamilton	✓	Yesterday at 16:36	
Jay	✓	Yesterday at 16:36	
Joint	✓	Yesterday at 18:04	
Madison	✓	Yesterday at 18:04	
Unknown	✓	Yesterday at 16:39	

As in the previous corpus, each folder represents the texts written by one author. The ‘Joint’ texts are coauthored by Madison and Hamilton. The 12 texts with disputed authorship are in the Unknown folder.

Data

First we load the corpus into R. Replace the directory below with the directory where you unzipped the files:

```
> M <- loadCorpus("~/Dropbox/Teaching/SCS/Data
  /Federalist/FunctionWords/", "frequentwords70")
> M$authornames
[1] "Hamilton" "Jay"      "Joint"    "Madison"  "Unknown"
```

So the unknown texts are in the 5th element of this list (Hamilton is first, Jay is second, etc).

Analysis

We create the training set by removing the unknown texts (element 5), and put these into the test set. Then we can classify as follows:

```
> train <- M$features[-5]
> test <- M$features[5]
> test <- test[[1]] #needed due to how the list is structured

#multinomial discriminant analysis, from stylometryfunctions.R
> discriminantCorpus(train,test)
 [1] 4 4 4 4 4 4 1 4 4 4 4 4

#KNN
> KNNCorpus(train,test)
 [1] 4 4 4 4 4 4 1 4 4 4 4 4
```

Comparing to the M$authornames field on the previous slide, author 4 is Madison and author 1 is Hamilton. Note tat both discriminant analysis and KNN have perfect agreement.

Analysis

We can check which of the Federalist Papers these correspond to using the M$booknames field (look at the 5th element, since the Unknown texts are in element 5 of the list)

```
> M$booknames
[[5]]
 [1] "49.txt" "50.txt" "51.txt" "52.txt" "53.txt" "54.txt"
 "55.txt" "56.txt" "57.txt" "58.txt" "62.txt" "63.txt"
```

So comparing to the previous slide, our prediction is that Federalist paper 55 was written by Hamilton and the other unknown papers were written by Madison.

Analysis

But can we trust these results? Perhaps stylometry methods don't work well on this corpus.

We should **validate** our methods by checking that they are capable of correctly predicting the author of the texts for which the true authorship is known.

In order words, for the texts that we know were definitely written by Hamilton/Madison/Jay, do discriminant analysis and KNN correctly predict the true known authors?

We can use leave-one-out cross validation for this:

Analysis

```
predictions <- NULL
KNNpredictions <- NULL
truth <- NULL
features <- M$features[-5] #discard unknown texts
for (i in 1:length(features)) {
  for (j in 1:nrow(features[[i]])) {
    testdata <- matrix(features[[i]][j,],nrow=1)
    traindata <- features
    traindata[[i]] <- traindata[[i]][-j,,drop=FALSE]

    pred <- discriminantCorpus(traindata, testdata)
    predictions <- c(predictions, pred)

    pred <- KNNCorpus(traindata, testdata)
    KNNpredictions <- c(KNNpredictions, pred)

    truth <- c(truth, i)
  }
}
```

Analysis

```
> sum(predictions==truth)/length(truth)
[1] 0.9863014
> sum(KNNpredictions==truth)/length(truth)
[1] 0.8493151
```

So discriminant analysis has 98.6% accuracy and KNN has 84.9% accuracy. We should probably trust discriminant analysis more. But fortunately they both agreed!

Analysis

```
> confusionMatrix(predictions,truth)
       predictions
targets  1  2  3  4
      1 51  0  0  0
      2  0  4  0  0
      3  0  0  3  0
      4  0  1  0 14
> confusionMatrix(KNNpredictions,truth)
       predictions
targets  1  2  3  4
      1 45  0  0  1
      2  2  2  0  0
      3  0  1  2  0
      4  4  2  1 13
```