Logistic Regression and Quadratic Discriminant Classification on Phoneme Data (in R)

Lucy L.

2024-01-06

This write-up is derived from the textbook "The Elements of Statistical Learning" from Chapter 5. Here I have reproduced figure 5.5. The dataset was extracted from the TIMIT database, a resource for speech recognition research. The dataset includes log-periodograms computed from 4509 speech frames, each 32 milliseconds long, representing five phonemes ("sh," "dcl," "iy," "aa," and "ao") from 50 male speakers.

First, I plot the log-periodogram as a function of frequency for 15 examples each of the phenomes "aa" and "ao" sampled from a total of 695 "aa"s and 1022 "ao"s. Each log-periodogram is measured at 256 uniformly spaced frequencies.

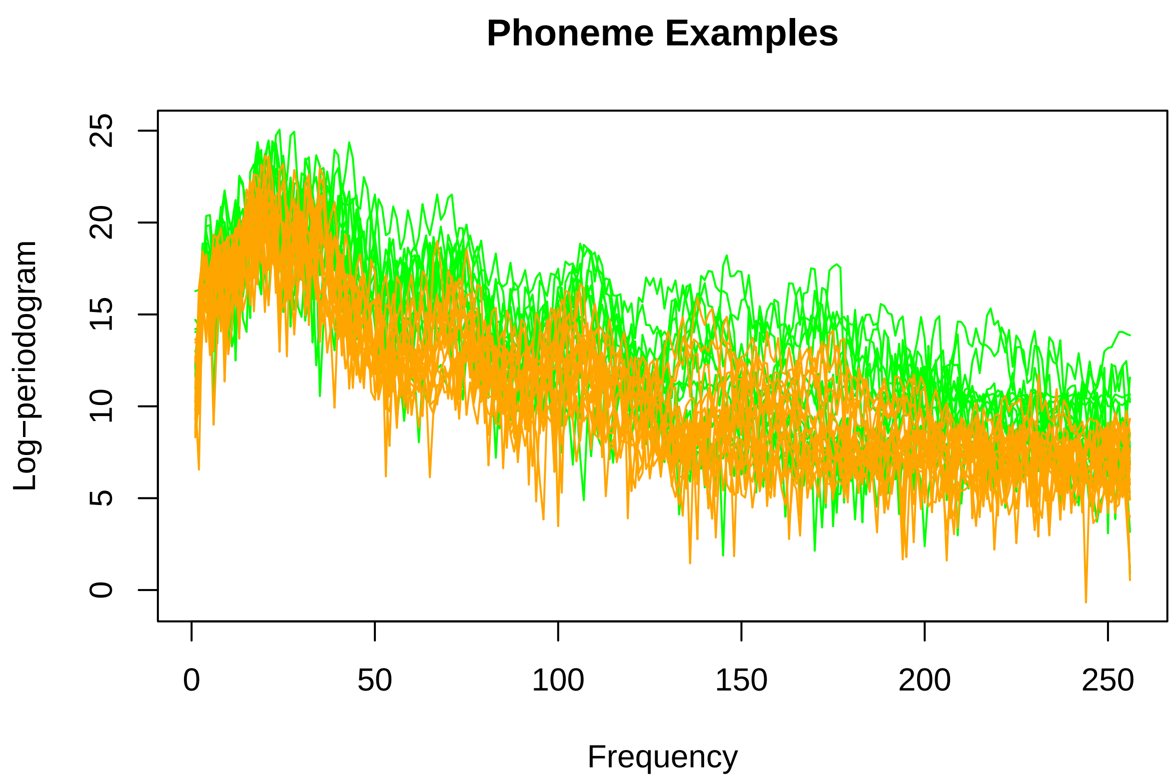
Second I plot the coefficients as a function of frequency of a logistic regression fit to the data by maximum likelihood, using the 256 log-periodogram values as inputs. The coefficients are restricted to be smooth in the green curve, and are unrestricted in the jagged black curve.

Third, I have examined the performance of trained logistic regression and quadratic discriminant classifiers on the data.

```r
library(MASS)
phoneme <- read.csv("phoneme.csv")
set.seed(0)
```

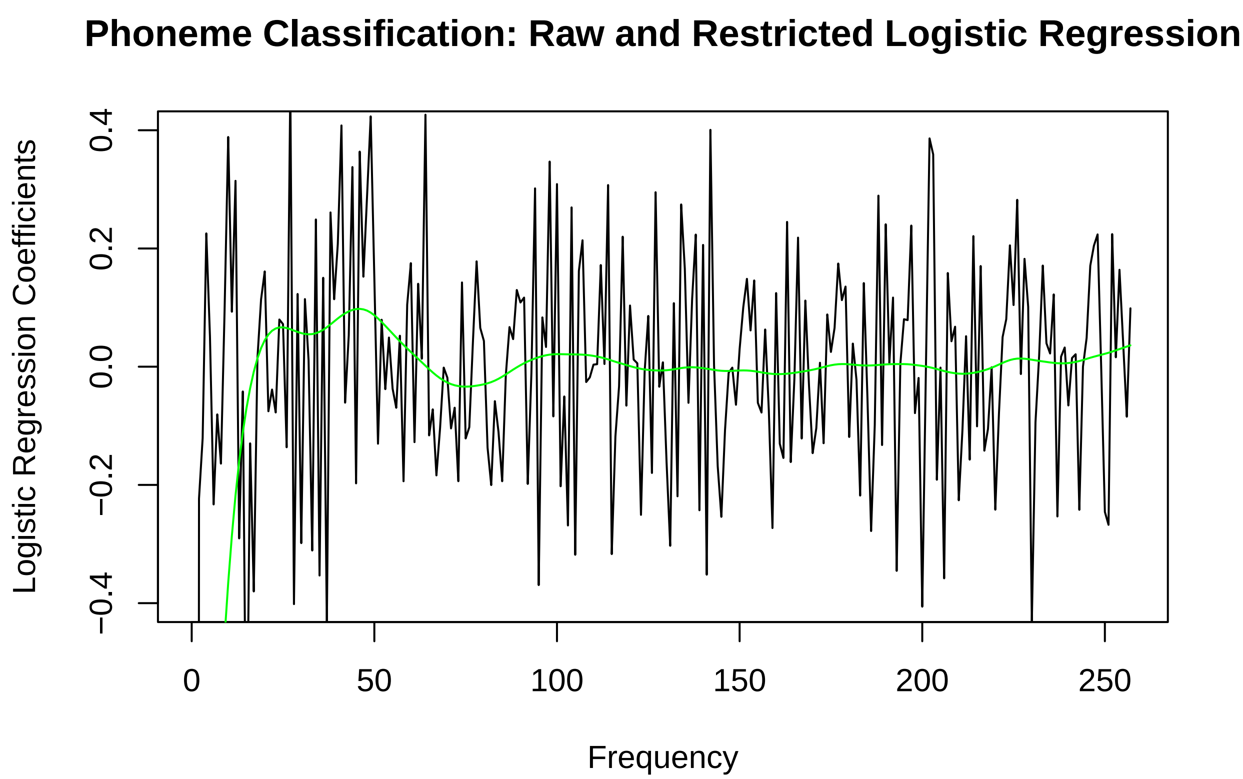
```r
aa_spot = phoneme$g == "aa"
ao_spot = phoneme$g == "ao"

aa_indx = which(aa_spot)
ao_indx = which(ao_spot)

# Plot of some examples of the two phonemes:
AA_data = phoneme[aa_indx[1:15], 1:256]
AO_data = phoneme[ao_indx[1:15], 1:256]

# Compute ylimits:
min_l = min(c(min(AA_data), min(AO_data)))
max_l = max(c(max(AA_data), max(AO_data)))

ii = 1
plot(as.double(AA_data[ii,]), ylim=c(min_l,max_l),
      type="l", col="green",
      xlab="Frequency", ylab="Log-periodogram",
      main = "Phoneme Examples")
for(ii in 2:dim(AA_data)[1]){
  lines(as.double(AA_data[ii,]), col="green")
```

```r
}
for(ii in 1:dim(AO_data)[1]){
  lines(as.double(AO_data[ii,]), col="orange")
}
```




```r
# Naive logistic regression classifier to classify aa's from ao's:

# Obtain training/testing data:
AA_data = phoneme[aa_indx,]
train_inds = grep("^train", AA_data$speaker)
test_inds = grep("^test", AA_data$speaker)
AA_data_train = AA_data[train_inds, 1:256]
AA_data_test = AA_data[test_inds, 1:256]

n_aa = dim(AA_data_train)[1]
AA_data_train$Y = rep(1, n_aa) # call this class 1
n_aa = dim(AA_data_test)[1]
AA_data_test$Y = rep(1, n_aa)

AO_data = phoneme[ao_indx,]
train_inds = grep("^train", AO_data$speaker)
test_inds = grep("^test", AO_data$speaker)
AO_data_train = AO_data[train_inds, 1:256]
AO_data_test = AO_data[test_inds, 1:256]
```

```r
n_ao = dim(AO_data_train)[1]
AO_data_train$Y = rep(0, n_ao) # call this class 0
n_ao = dim(AO_data_test)[1]
AO_data_test$Y = rep(0, n_ao)

DT_train = rbind(AA_data_train, AO_data_train)
DT_test = rbind(AA_data_test, AO_data_test)

form = paste("Y ~", paste( colnames(DT_train)[1:256], collapse="+"))
m = glm(form, family=binomial, data=DT_train)

# Plot the coefficients from logistic regression fit
mc = as.double( coefficients(m)[-1] )
mc = as.double( coefficients(m) )
plot(mc, ylim=c(-0.4,+0.4), type="l", xlab="Frequency",
      ylab="Logistic Regression Coefficients",
      main = "Phoneme Classification: Raw and Restricted Logistic Regression")

lines(smooth.spline(mc,df=15), col="green")
```




```r
# Checking trained logistic regression classifier performance:
Y_hat_train = predict(m, DT_train[,1:256], type="response")
predicted_class_label_train = as.double(Y_hat_train > 0.5)

Y_hat_test  = predict(m, DT_test[,1:256], type="response")
```

```r
predicted_class_label_test = as.double(Y_hat_test > 0.5)

err_rate_train = mean(DT_train[,257] != predicted_class_label_train)
err_rate_test  = mean(DT_test[,257] != predicted_class_label_test)
print( sprintf('Logistic Regression: Training error rate = %10.6f; Test error rate = %10.5f', err_rate_t
```

```
## [1] "Logistic Regression: Training error rate =   0.093114; Test error rate =    0.24374"
```

```r
# Checking quadratic discriminant classifier performance:
qdam = qda( DT_train[,1:256], DT_train[,257])
predTrain = predict(qdam, DT_train[,1:256])
predTest  = predict(qdam, DT_test[,1:256])

err_rate_train = mean(predTrain$class != DT_train[,257])
err_rate_test  = mean(predTest$class != DT_test[,257])
print(sprintf('Quadradic Discriminant: Training error rate = %10.6f; Test error rate = %10.5f', err_rat
```

```
## [1] "Quadradic Discriminant: Training error rate =   0.000000; Test error rate =    0.33941"
```