Project 2 Supplemental Information on Quadratic Logistic Regression

Shengdong Zhang
Department of Computing Science
Simon Fraser University
Burnaby, BC Canada V5A 1S6
`sza75@sfu.ca`

Maximizing Discriminant Functions

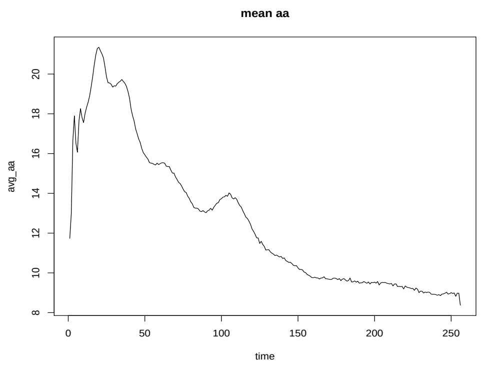
Logistic regression classifiers (LR), unlike LDA or QDA, estimates posterior probabilities $P(G = k|X = \vec{x})$. It directly learns parameters $\vec{\beta}_i \, \forall i = 1, \ldots, K$ from given samples. Once it is trained, the term inside the exponential function corresponding to the ith class, namely, $\vec{\beta}_i^T \vec{x} + \beta_{i0}$, is the discriminant function $g_i(\vec{x})$ of the ith class. So the input $\vec{x}$ which can maximize $g_i(\vec{x})$ is the learned feature. However, since $g_i(\vec{x})$ can be increased to infinity by increasing the norm of $\vec{x}$ limitlessly, we need to find a input pattern $\vec{x}$ with unit norm to represent the feature. Obviously, the learned feature for $g_i(\vec{x})$ is $\vec{x}_{opt}^{(i)} = \vec{\beta}_i / ||\vec{\beta}_i||$. Similarly, for Quadratic logistic regression classifiers (QLR), we have the discriminant function for ith class $\vec{x}^T W_i \vec{x} + \vec{\beta}_i^T \vec{x} + \beta_{i0}$. Thus, the learned feature for $g_i(\vec{x})$ in this case is:

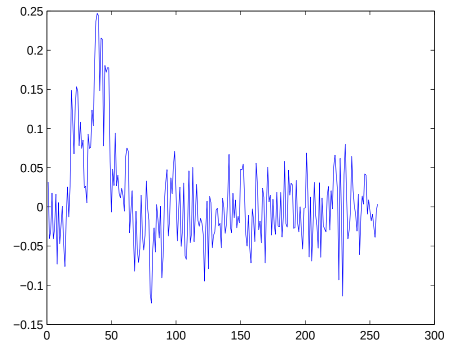
$$\vec{x}_{opt}^{(i)} = \arg\max \; \vec{x}^T W_i \vec{x} + \vec{\beta}_i^T \vec{x} + \beta_{i0} \quad \text{s.t. } \vec{x}^T \vec{x} = 1$$

It is a quadratic programming problem with a quadratic constraint. There are many packages available to solve this problem. In my project, I used **fmincon** function in MATLAB.

The mean signal and the leaned features of LR will be shown below for each phoneme class first, then are the corresponding learned features of QLR with L2 penalty and QLR with additive Gaussian noise for comparison. We can see that both LR and QLR with Gaussian additive noise learned similar features, and the features learned by QLR with L2 penalty are much less wiggly than the others.

Phoneme aa





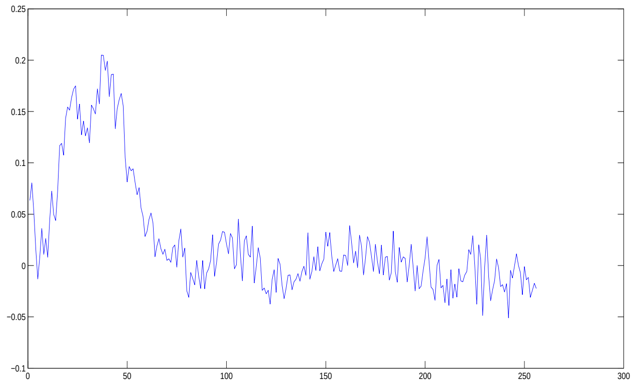


Figure 1: Feature Learned By QLR with L2 Penalty

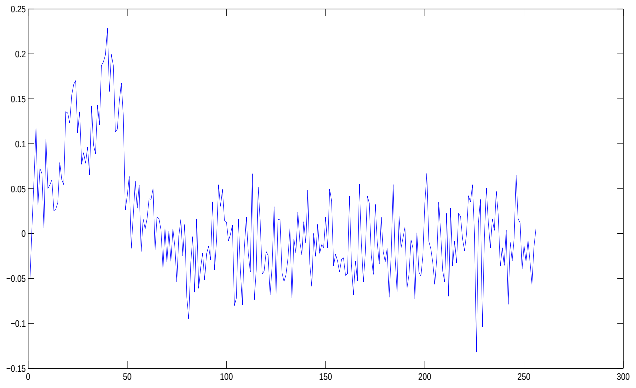
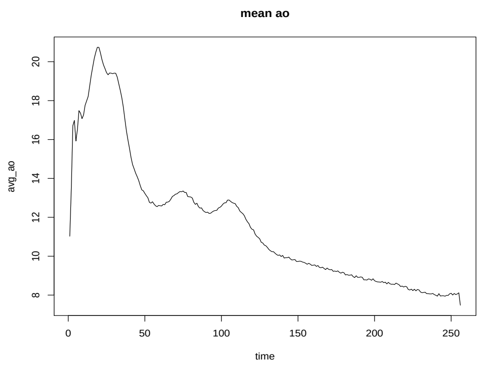
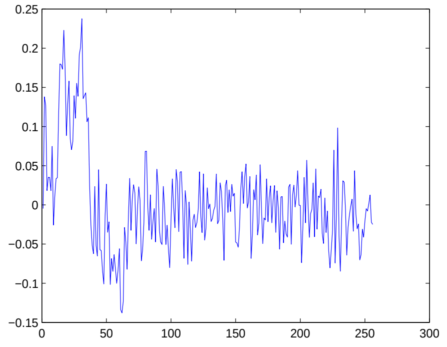


Figure 2: Feature Learned By QLR with Additive Gaussian Noise

Phoneme ao





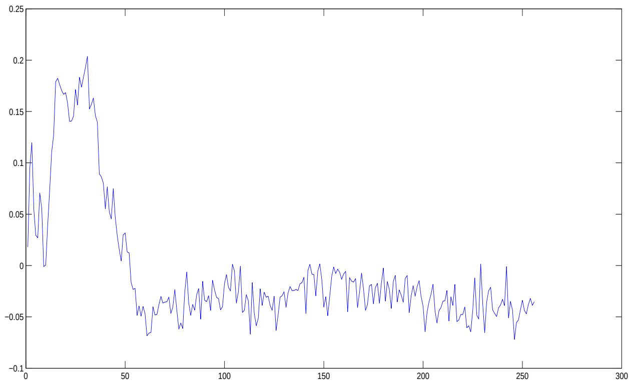


Figure 3: Feature Learned By QLR with L2 Penalty

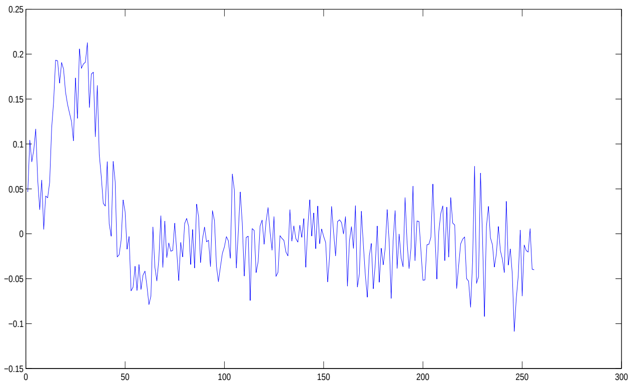
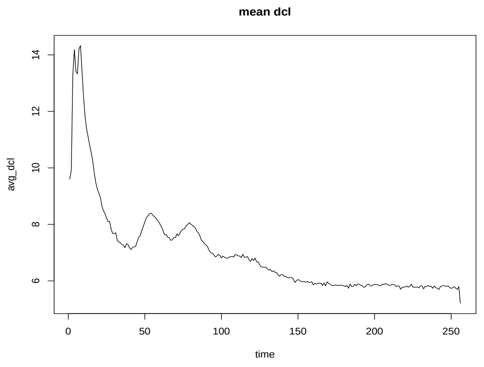
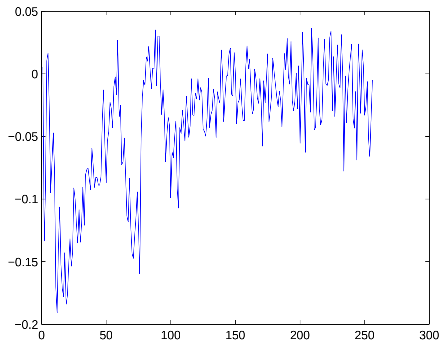


Figure 4: Feature Learned By QLR with Additive Gaussian Noise

Phoneme dcl





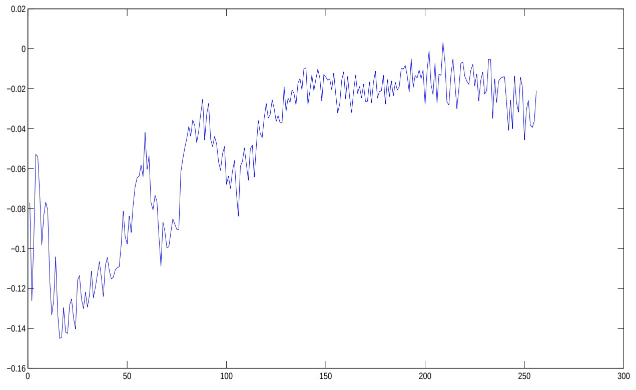


Figure 5: Feature Learned By QLR with L2 Penalty

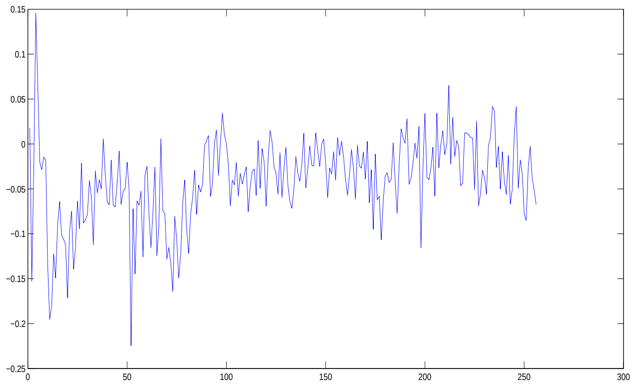
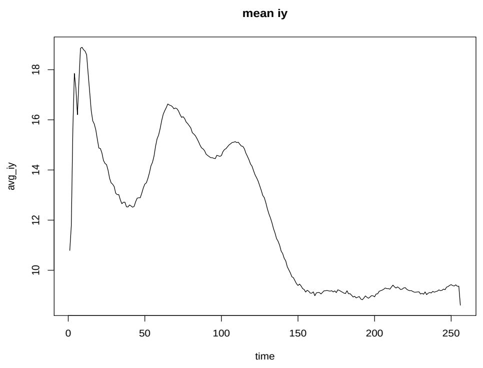
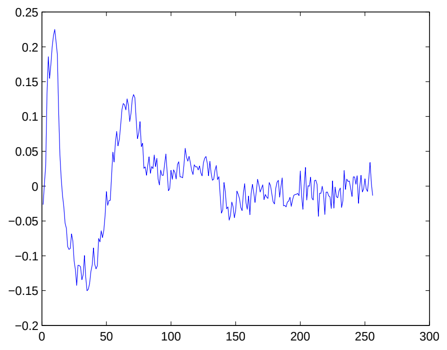


Figure 6: Feature Learned By QLR with Additive Gaussian Noise

Phoneme iy





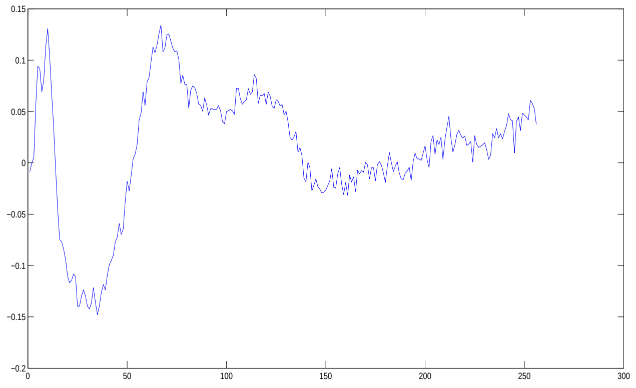


Figure 7: Feature Learned By QLR with L2 Penalty

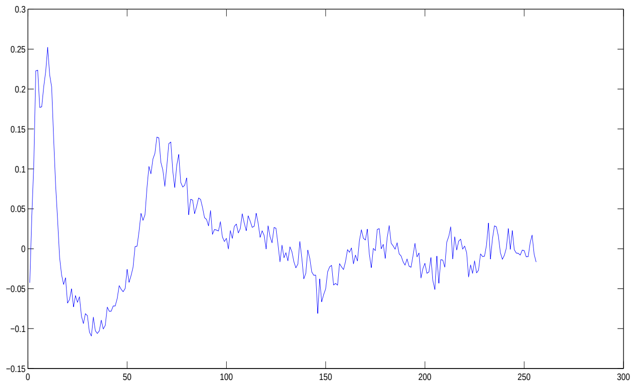
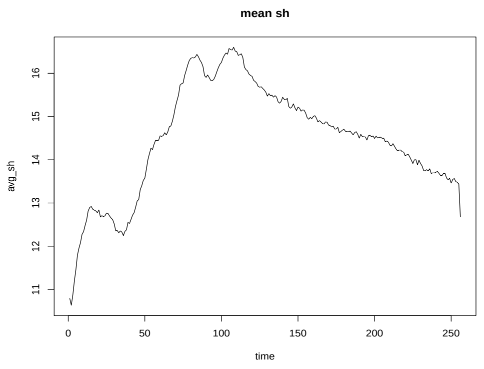
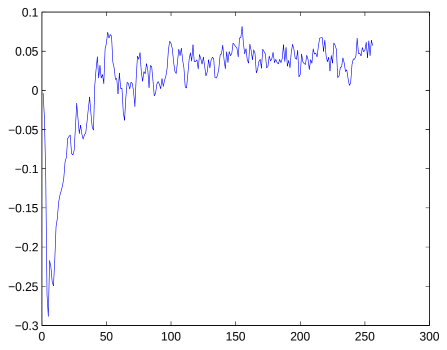


Figure 8: Feature Learned By QLR with Additive Gaussian Noise

Phoneme sh





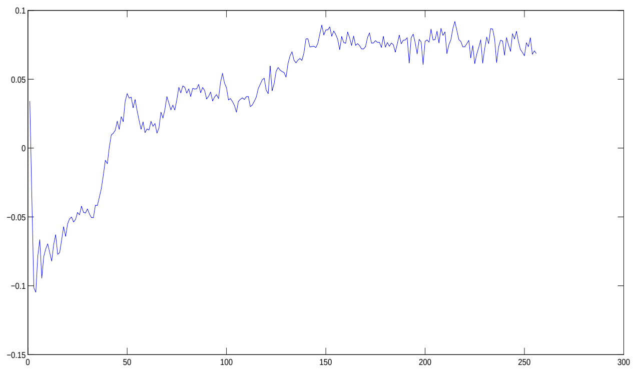


Figure 9: Feature Learned By QLR with L2 Penalty

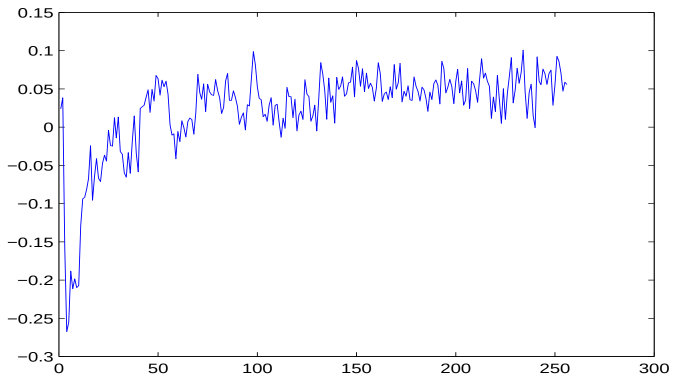


Figure 10: Feature Learned By QLR with Additive Gaussian Noise

Confusion Matrices Comparison

Because the results on different resamples are similar, only the first one is taken to discuss. Below are the test confusion matrices given by LR and QLR with Gaussian Noise on the first resample. From the plots for phoneme classes "aa" and "ao", we know that these two classes are similar to each other. Compared to LR, QLRwGN seems to overfit the data of these two classes. For the other three classes, both classifiers have similar performance.

```
LR
true\ pred   aa     ao     dcl     iy     sh
aa          191     73       0      0      0
ao           48    334       1      0      0
dcl           0      0     259      2      1
iy            0      0       0    433      1
sh            0      0       0      0    312

QLR with Gaussian Noise
true\ pred   aa     ao     dcl     iy     sh
aa          192     69       1      2      0
ao           77    305       0      1      0
dcl           0      0     259      3      0
iy            1      0       0    432      1
sh            0      0       0      0    312
```