## Problems

1. LFD Coding
2. LFD Problem 3.19
3. LFD Exercise 4.5
4. LFD Problem 4.8

# Q1: Gradient Descent

| Number of Iteration | $E_{in}$ | Training Error | Testing Error |
|---|---|---|---|
| 1000000 | 0.4354 | 0.1513 | 0.1310 |
| 100000 | 0.4937 | 0.2237 | 0.2069 |
| 10000 | 0.5847 | 0.3092 | 0.3172 |
| GLM | 0.4074 | 0.1711 | 0.1103 |

The table above summarize the $E_{in}$, training and testing error for required number of iteration as well as glm function. The default step size for all number of iteration is $1e^{-5}$.

It looks like the both $E_{in}$ and Errors decrease when we increase the step size, which suggests logistic regression has good generalization property and it does not "over-fit" the data when we increase the maximum iteration. Glmfit, compare to our algorithm, runs better (provide smaller $E_{in}$ and classification error) and faster. Although if keeping increasing the max number of iterations, the result will get closer to that of glm but the time taken for computation will be extremely high.

| Step Size | Number of Iteration before termination | $E_{in}$ |
|---|---|---|
| 10 | 1000000 | 0.4354 |
| 1 | 248 | 0.4074 |
| $10^{-1}$ | 2514 | 0.4074 |
| $10^{-2}$ | 25172 | 0.4074 |
| $10^{-3}$ | 251748 | 0.4074 |
| $10^{-4}$ | 1000000 | 0.4074 |

The table above report the number of iteration and $E_{in}$ for different step size. The maximum number of iteration is still set to 1000000 to avoid non-terminating condition. If the learning rate is too large, we can see the algorithm can hardly converge and therefore fail to terminate before the max iteration and gives larger $E_{in}$ compared to other runs. For the rest of runs, the larger the step size is, the quicker the algorithm converges and all of runs result in same $E_{in}$ (or same w). And if the step size is too small, the number of iteraion could also exceeds the set limit.

# Q2: LFD Problem 3.19

### (a)

One problem with this feature transform is that it will map all testing (or real data) which is not in the training set to (0, 0, 0,..., 0). That would make originally different data points not distinguishable after feature transformation.

### (b)

The value of the feature transform decreases with distance and ranges between zero (in the limit) and one (when x = x'), so it has a ready interpretation as a similarity measure and also avoided the problem in part (a). It also map the original data to higher dimensional space at which the data will have higher probability to be linear separable. However, it does not scale well to large numbers of training samples input space since it will result in computation in extremely high dimensional space.

### (c)

Although it solves the problem that the computational cost at higher space but since the mapping is limited to $100^2$ dimensions. Very complex data may not be linear separable at limited dimension.

# Q3: LFD Exercise 4.5

## (a)

$\Gamma$ could be any matrix satisfy $\Gamma^T\Gamma = I$, one example is the Q*Q dimensional identity matrix.

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

## (b)

$\Gamma$ could be any matrix satisfy $\Gamma^T\Gamma$ equals to Q*Q dimensional matrix with all one, One example is the Q*Q dimensional matrix whose first column is all ones and rest are all zeros.

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

## Q4: LFD Problem 4.8

$\nabla \mathbb{E}_{aug}(w) = \nabla \mathbb{E}_{in}(w) + 2\lambda w$

So,

$$\begin{aligned} w(t+1) &\leftarrow w(t) - \eta\nabla(\mathbb{E}_{aug}(w(t)) \\ w(t+1) &\leftarrow w(t) - \eta(\nabla \mathbb{E}_{in}(w) + 2\lambda w(t)) \\ w(t+1) &\leftarrow w(t) - 2\eta\lambda w(t) - \eta\nabla \mathbb{E}_{in}(w) \\ w(t+1) &\leftarrow (1 - 2\eta\lambda)w(t) - \eta\nabla \mathbb{E}_{in}(w) \end{aligned}$$