Jiafeng Ni
CSCI567
Homework #2
02/26/2016

1. Naive Bayes, 2. Generative Model and Discriminative Model, 3. Multinomial Logistic Regression, 4. Programming Problems

1.(a) Suppose you are given the data for several 3-digit passwords and the output whether they open a security gate or not. The passwords are generated by a source with following dis- tribution. Compute the parameters (priors and conditionals) of a multinomial Naive Bayes classifier that uses the digits 0, 1, 2, 3, and 4 as features assuming smoothing rule that the computed-zero probabilities are smoothed into probability 0.01 instead. You are not required to correct $P(X) + P(X') > 1$ which might be caused because of the smoothing rule.

(1.*a*) Based on the statistics given in the table, we have the priors and conditionals as follows:

$$P(open) = \frac{1}{18} + \frac{1}{18} = \frac{1}{9} \qquad P(\neg open) = 1 - P(open) = \frac{8}{9}$$
$$P(0 \mid open) = \frac{1}{3+3} = \frac{1}{6} \qquad P(0 \mid \neg open) = \frac{1}{3+3} = \frac{1}{6}$$
$$P(1 \mid open) = \frac{1}{3+3} = \frac{1}{6} \qquad P(1 \mid \neg open) = \frac{0}{3+3} = \frac{1}{100}$$
$$P(2 \mid open) = \frac{2}{3+3} = \frac{1}{3} \qquad P(2 \mid \neg open) = \frac{1}{3+3} = \frac{1}{6}$$
$$P(3 \mid open) = \frac{1}{3+3} = \frac{1}{6} \qquad P(3 \mid \neg open) = \frac{2}{3+3} = \frac{1}{3}$$
$$P(4 \mid open) = \frac{1}{3+3} = \frac{1}{6} \qquad P(4 \mid \neg open) = \frac{2}{3+3} = \frac{1}{3}$$

■

1.(b) Consider the password 031, will it open the gate based on your classifier?

(1.*b*) Taking advantage of what we get in problem 1.a, we have:

$$P(open \mid 031) = \frac{P(031 \mid open) \cdot P(open)}{P(031)}$$

$$P(\neg open \mid 031) = \frac{P(031 \mid \neg open) \cdot P(\neg open)}{P(031)}$$

And we also know that:

$$P(open \mid 031) + P(\neg open \mid 031) = 1$$

$$P(031 \mid open) \cdot P(open) = \frac{1}{6} \cdot \frac{1}{6} \cdot \frac{1}{6} \cdot \frac{1}{9}$$

$$P(031 \mid \neg open) \cdot P(\neg open) = \frac{1}{3} \cdot \frac{1}{100} \cdot \frac{1}{6} \cdot \frac{8}{9}$$

So, $\frac{P(031 \mid open) \cdot P(open)}{P(031 \mid \neg open) \cdot P(\neg open)} = \frac{25}{24}$, i.e. $P(open \mid 031) > 0.5$. Thus, it will open the gate.

■

1.(c) How can we show that *Logistic Regression* is the discriminative version of *Naive Bayes*?

(1.*c*) Let's consider binary logistic regression, but we first start from *Naive Bayes*. Suppose $Y \sim Bernollo(\pi)$, and $P(X_i \mid Y = y_k)$ is *Guassian*. According to Bayes rule and marginal distribution, we have:

$$\begin{aligned} P(Y = 1 \mid X) &= \frac{P(Y = 1) \cdot P(X \mid Y = 1)}{P(Y = 1) \cdot P(X \mid Y = 1) + P(Y = 0) \cdot P(X \mid Y = 0)} \\ &= \frac{1}{1 + \frac{P(Y=0) \cdot P(X \mid Y=0)}{P(Y=1) \cdot P(X \mid Y=1)}} \\ &= \frac{1}{1 + exp\left[\ln \frac{P(Y=0) \cdot P(X \mid Y=0)}{P(Y=1) \cdot P(X \mid Y=1)}\right]} \\ &= \frac{1}{1 + exp\left[\ln \frac{P(Y=0)}{P(Y=1)} + \sum_i \ln \frac{P(X_i \mid Y=0)}{P(X_i \mid Y=1)}\right]} \end{aligned} \tag{1}$$

Now consider just the summation in the denominator of equation (1). Given our assumption that $P(X_i \mid Y = y_k)$ is *Gaussian*, we can expand this term as:

$$\begin{aligned} \sum_i \ln \frac{P(X_i \mid Y = 0)}{P(X_i \mid Y = 1)} &= \sum_i \ln \frac{\frac{1}{\sqrt{2\pi}\sigma_i} exp\left[-\frac{(X_i - \mu_{0,i})^2}{2{\sigma_i}^2}\right]}{\frac{1}{\sqrt{2\pi}\sigma_i} exp\left[-\frac{(X_i - \mu_{1,i})^2}{2{\sigma_i}^2}\right]} \\ &= \sum_i \ln\ exp\left[-\frac{(X_i - \mu_{o,i})^2 - (X_i - \mu_{1,i})^2}{2{\sigma_i}^2}\right] \\ &= \sum_i \left(\frac{\mu_{0,i} - \mu_{1,i}}{{\sigma_i}^2} X_i + \frac{{\mu_{1,i}}^2 - {\mu_{0,i}}^2}{2{\sigma_i}^2}\right) \end{aligned} \tag{2}$$

Substituting (2) into (1):

$$P(Y = 1 \mid X) = \frac{1}{1 + exp\left[\ln \frac{1-\pi}{\pi} + \sum_i \left(\frac{\mu_{0,i} - \mu_{1,i}}{{\sigma_i}^2} X_i + \frac{{\mu_{1,i}}^2 - {\mu_{0,i}}^2}{2{\sigma_i}^2}\right)\right]} \tag{3}$$

or equivalently:

$$P(Y = 1 \mid X) = \frac{1}{1 + exp(\omega_0 + \sum_i \omega_i \cdot X_i)} \tag{4}$$

where the weights $\omega_1, \omega_2, \ldots, \omega_n$ are given by:

$$\omega_i = \frac{\mu_{0,i} - \mu_{1,i}}{{\sigma_i}^2} \tag{5}$$

and where:

$$\omega_0 = \ln \frac{1 - \pi}{\pi} + \sum_i \frac{{\mu_{1,i}}^2 - {\mu_{0,i}}^2}{2{\sigma_i}^2} \tag{6}$$

The form of the equation (4) is exactly the same as the Logistic Regression. Similar result can be proved if Y is multivariate. Thus we can say that *Logistic Regression* is the discriminative version of *Naive Bayes* because it considers conditional probability, and can be derived from *Naive Bayes* given additional assumptions.

And also, *Logistic Regression* is a function approximation algorithm that uses training data to directly estimate $P(Y \mid X)$, in contrast to *Naive Bayes*. In this sense, *Logistic Regression* is often referred to as a discriminative classifier because we can view the distribution $P(Y \mid X)$ as directly discriminating the value of the target value Y for any given instance X.

■

1.(d) The classifier we learn by optimizing the Naive Bayes conditional likelihood is nevertheless not the same as what we would have learned for a Logistic Classifier directly. What modeling assumption makes it somewhat less generic than Logistic Regression?

(1.*d*) The *Naive Bayes* makes the assumption that features in observed values are conditional independent given the class of it, which makes it less generic than *Logistic Regression*.

■

2.(a) Please write out the log-likelihood of distribution $p(D \mid \pi, \mu_0, \mu_1, S_0, S_1)$, and the MLE of five parameters in this distribution. For conciseness, we use(1)$t_{n,c}$ as 0/1 indicator that sample is of class c, (2) N_c as total count of samples from class c.

(2.*a*) Given the distributions in the problem statement, the joint probability would be:

$$p(x_n, y_n) = \pi^{t_{n,1}} \cdot (1 - \pi)^{t_{n,0}} \cdot \left[\frac{exp[-\frac{1}{2}(x_n - \mu_0)^T \Sigma_0^{-1} (x_n - \mu_0)]}{(2\pi)^{D/2} |\Sigma_0|^{1/2}} \right]^{t_{n_0}} \cdot \left[\frac{exp[-\frac{1}{2}(x_n - \mu_1)^T \Sigma_1^{-1} (x_n - \mu_1)]}{(2\pi)^{D/2} |\Sigma_1|^{1/2}} \right]^{t_{n_1}} \tag{7}$$

So, the log-likelihood of distribution is:

$$\begin{aligned} p(D) &= \log \prod_{n=1}^{N} p(x_n, y_n) \\ &= \sum_{n=1}^{N} \{ t_{n,1} \log \pi + t_{n,0} \log(1-\pi) - \\ &\qquad t_{n,0} \left[\frac{D}{2} \log 2\pi + \frac{1}{2} \log |\Sigma_0| + \frac{1}{2} (x_n - \mu_0)^T \Sigma_0^{-1} (x_n - \mu_0) \right] - \\ &\qquad t_{n,1} \left[\frac{D}{2} \log 2\pi + \frac{1}{2} \log |\Sigma_1| + \frac{1}{2} (x_n - \mu_1)^T \Sigma_1^{-1} (x_n - \mu_1) \right] \} \end{aligned} \tag{8}$$

Additionally, we know that features in x are conditional independent given y, thus we have:

$$\log |\Sigma_0| = \sum_{i=1}^{D} \log \Sigma_{0,i} \tag{9}$$

$$\log |\Sigma_1| = \sum_{i=1}^{D} \log \Sigma_{1,i} \tag{10}$$

$$(x_n - \mu_0)^T \Sigma_0^{-1} (x_n - \mu_0) = \sum_{i=1}^{D} \frac{(x_{n,i} - \mu_{0,i})^2}{\Sigma_{0,i}} \tag{11}$$

$$(x_n - \mu_1)^T \Sigma_1^{-1} (x_n - \mu_1) = \sum_{i=1}^{D} \frac{(x_{n,i} - \mu_{1,i})^2}{\Sigma_{1,i}} \tag{12}$$

Substituting (9), (10), (11), and (12) back into (8), we can get the partial dirivatives as:

$$\frac{\partial p(D)}{\partial \pi} = \sum_{n=1}^{N} (\frac{t_{n,1}}{\pi}) - \frac{t_{n,0}}{1-\pi} = \frac{N_1}{\pi} - \frac{N_0}{1-\pi} = 0 \tag{13}$$

$$\frac{\partial p(D)}{\partial \mu_{0,i}} = \sum_{n=1}^{N} t_{n,0} \cdot \frac{x_{n,i} - \mu_{0,i}}{\Sigma_{0,i}} = -\frac{N_0}{\Sigma_{0,i}} \mu_{0,i} + \sum_{n:y=0} \frac{x_{n,i}}{\Sigma_{0,i}} = 0 \tag{14}$$

$$\frac{\partial p(D)}{\partial \mu_{1,i}} = \sum_{n=1}^{N} t_{n,1} \cdot \frac{x_{n,i} - \mu_{1,i}}{\Sigma_{1,i}} = -\frac{N_1}{\Sigma_{1,i}} \mu_{1,i} + \sum_{n:y=1} \frac{x_{n,i}}{\Sigma_{1,i}} = 0 \tag{15}$$

$$\frac{\partial p(D)}{\partial \Sigma_{0,i}} = \sum_{n=1}^{N} -t_{n,0} \left(\frac{1}{2\Sigma_{0,i}} - \frac{(x_{n,i} - \mu_{0,i})^2}{2{\Sigma_{0,i}}^2} \right) = \frac{1}{2{\Sigma_{0,i}}^2} [-N_0 \Sigma_{0,i} + \sum_{n:y=0} (x_{n,i} - \mu_{0,i})^2] \tag{16}$$

$$\frac{\partial p(D)}{\partial \Sigma_{1,i}} = \sum_{n=1}^{N} -t_{n,1} \left(\frac{1}{2\Sigma_{1,i}} - \frac{(x_{n,i} - \mu_{1,i})^2}{2{\Sigma_{1,i}}^2} \right) = \frac{1}{2{\Sigma_{1,i}}^2} [-N_1 \Sigma_{1,i} + \sum_{n:y=1} (x_{n,i} - \mu_{1,i})^2] \tag{17}$$

Solve the equations (13)$\tilde{}$(17), we can get the MLE of five parameters as follows:

$$\pi = \frac{N_1}{N_0 + N_1} = \frac{N_1}{N} \tag{18}$$

$$\mu_0 = \frac{1}{N_0} \sum_{n:y=0} x_n \tag{19}$$

$$\mu_0 = \frac{1}{N_1} \sum_{n:y=1} x_n \tag{20}$$

$$\Sigma_{0,i} = \frac{1}{N_0} \sum_{n:y=0} (x_{n,i} - \mu_{0,i})^2 \tag{21}$$

$$\Sigma_{1,i} = \frac{1}{N_1} \sum_{n:y=1} (x_{n,i} - \mu_{1,i})^2 \tag{22}$$

It's easy to be seen that MLE of π is the expectation of y, MLE of μ is the mean of x with the corresponding label, MLE of each dimension of Σ is the variance of x with the corresponding label.

■

2.(b) According to Bayesian rule, after we have estimated parameters, we can infer the posterior probability of class given observation:

$$p(y = 1 \mid x, \pi, \mu_0, \mu_1, S_0, S_1) = \frac{p(y = 1 \mid \pi) p(x \mid y, \mu_1, S_1)}{p(y = 1 \mid \pi) p(x \mid y, \mu_1, S_1) + p(y = 0 \mid \pi) p(x \mid y, \mu_0, S_0)}$$

Please prove that *iff* (if and only if) $S_0 = S_1$, the posterior distribution can be simplified into generalized linear model:

$$p(y = 1 \mid x, \pi, \mu_0, \mu_1, S_0, S_1) = \frac{1}{1 + exp(-\theta^T x)}$$

And please write down the expression of θ when $S_0 = S_1$.

(2.*b*) According to Bayes Rule, and the distribution given in the problem statement, we get:

$$\begin{aligned} p(y = 1 \mid x) &= \frac{p(y = 1) \cdot p(x \mid y = 1)}{p(y = 1) \cdot p(x \mid y = 1) + p(y = 0) \cdot p(x \mid y = 0)} \\ &= \frac{1}{1 + \frac{p(y=0) \cdot p(x \mid y=0)}{p(y=1) \cdot p(x \mid y=1)}} \\ &= \frac{1}{1 + \frac{1-\pi}{\pi} \frac{\frac{1}{(2\pi)^{D/2} |\Sigma_0|^{1/2}} exp\left[-\frac{1}{2}(x-\mu_0)^T \Sigma_0^{-1} (x-\mu_0)\right]}{\frac{1}{(2\pi)^{D/2} |\Sigma_1|^{1/2}} exp\left[-\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1} (x-\mu_1)\right]}} \\ &= \frac{1}{1 + exp\left\{ \ln \frac{1-\pi}{\pi} + \frac{1}{2} \ln \frac{|\Sigma_1|}{|\Sigma_0|} - \frac{1}{2} \sum_{i=1}^{D} \left[\frac{(x_i - \mu_{1,i})^2}{\Sigma_{1,i}} - \frac{(x_i - \mu_{0,i})^2}{\Sigma_{0,i}} \right] \right\}} \end{aligned} \tag{23}$$

From equation (23), we can see that the posterior distribution becomes a generalized linear model of x if and only if $\Sigma_{0,i} = \Sigma_{1,i} = \Sigma_i$ holds. And based on that, we can get:

$$\frac{(x_i - \mu_{1,i})^2}{\Sigma_{1,i}} - \frac{(x_i - \mu_{0,i})^2}{\Sigma_{0,i}} = \frac{2(\mu_{0,i} - \mu_{1,i})x + (\mu_{1,i}^2 - \mu_{0,1}^2)}{\Sigma_i} \tag{24}$$

Substituting equation (24) back into (23), we have:

$$p(y = 1 \mid x) = \frac{1}{1 + exp\{ \ln \frac{1-\pi}{\pi} + \frac{1}{2} \sum_{i=1}^{D} \frac{2(\mu_{0,i} - \mu_{1,i})x + (\mu_{1,i}^2 - \mu_{0,1}^2)}{\Sigma_i} \}} \tag{25}$$

or equivalently:

$$p(y = 1 \mid x) = \frac{1}{1 + exp(-\theta_0 - \sum_{i=1}^{D} \theta_i \cdot x_i)} \tag{26}$$

where the weights are:

$$\theta_0 = -\ln \frac{1-\pi}{\pi} - \sum_{i=1}^{D} \frac{\mu_{1,i}^2 - \mu_{0,i}^2}{2\Sigma_i} \tag{27}$$

$$\theta_i = \frac{\mu_{0,i} - \mu_{1,i}}{\Sigma_i} \tag{28}$$

Thus, based on equations (26), (27), and (28) we can write the posterior distribution just as the form as *Logistic Regression* is, which of course is a GLM.

■

2.(c) Instead of mixture of Gaussian distribution, suppose data are generated from mixture of Poisson distributions. In this case, the observation x has discrete integer value:

$$x|y = 0 \ \sim \ Poisson(\lambda_0)$$

$$x|y = 1 \ \sim \ Poisson(\lambda_1)$$

Please prove that posterior distribution $p(y|x, \pi, \lambda_0, \lambda_1)$ can be modeled with generalized linear model.

(2.*c*) According to the conditional distribution, we have:

$$p(x \mid y = 0) = \frac{\lambda_0^x e^{-\lambda_0}}{x!} \tag{29}$$

$$p(x \mid y = 1) = \frac{\lambda_1^x e^{-\lambda_1}}{x!} \tag{30}$$

and using Bayes Rule, we get:

$$\begin{aligned} p(y \mid x) &= \frac{p(x,y)}{p(x \mid y=1)p(y=1) + p(x \mid y=0)p(y=0)} \\ &= \frac{\pi^y(1-\pi)^{1-y}\left(\frac{\lambda_1^x e^{-\lambda_1}}{x!}\right)^y \left(\frac{\lambda_0^x e^{-\lambda_0}}{x!}\right)^{1-y}}{\pi\frac{\lambda_1^x e^{-\lambda_1}}{x!} + (1-\pi)\frac{\lambda_0^x e^{-\lambda_0}}{x!}} \\ &= \frac{\pi^y(1-\pi)^{1-y}(\lambda_1^x e^{-\lambda_1})^y(\lambda_0^x e^{-\lambda_0})^{1-y}}{\pi\lambda_1^x e^{-\lambda_1} + (1-\pi)\lambda_0^x e^{-\lambda_0}} \\ &= \frac{1}{\left[\frac{\pi}{1-\pi}e^{\lambda_0-\lambda_1}\left(\frac{\lambda_1}{\lambda_0}\right)^x\right]^{1-y} + \left[\frac{\pi}{1-\pi}e^{\lambda_0-\lambda_1}\left(\frac{\lambda_1}{\lambda_0}\right)^x\right]^{-y}} \end{aligned} \tag{31}$$

thus, expectation of y given x is:

$$\mathbb{E}(y) = \frac{1}{1 + \frac{1-\pi}{\pi}e^{\lambda_1-\lambda_0}\left(\frac{\lambda_0}{\lambda_1}\right)^x} \tag{32}$$

Notice that both (31) and (32) are generalized model as to x, thus $p(y \mid x)$ can be modeled wit GLM. Proof done.

■

3. Please prove that at the optimum we have $\sum_{k=1}^{C} \hat{\omega}_{k,j} = 0$ for $j = 1 : D$. (For unregularized terms $\hat{\omega_{k0}}$, we still need to enforce that $\omega_{0,C} = const$ to ensure identifiability of the offset.)

(3) Given the conditional probability of y on x and W, we get the log of it as:

$$\log p(y = c \mid x, W) = \omega_{c0} + \omega_c^T x - \sum_{k=1}^{C} exp(\omega_{k0} + \omega_k^T x) \tag{33}$$

Based on equation (33), we can get the equation we want to optimize as follows:

$$equation = \sum_{k=1}^{C} \sum_{n:y_n=k} (\omega_{k0} + \omega_k^T x_n) - \sum_{n=1}^{N} \log[\sum_{k=1}^{C} exp(\omega_{k0} + \omega_k^T x_n)] - \lambda \sum_{k=1}^{C} ||\omega_k||_2^2 \tag{34}$$

Then, the partial derivative of this equation over ω_c would be:

$$\frac{\partial\, equation}{\omega_k} = \sum_{n:y_n=k} x_n - \sum_{n=1}^{N} \frac{exp(\omega_{k0} + \omega_k^T x_n)}{\sum_{k'=1}^{C} exp(\omega_{k'0} + \omega_{k'}^T x_n)} x_n - \lambda\omega_k \tag{35}$$

To optimize (34), we require (35) to be equal to 0. Thus, we have:

$$\lambda\omega_k = \sum_{n:y_n=k} x_n - \sum_{n=1}^{N} \frac{exp(\omega_{k0} + \omega_k^T x_n)}{\sum_{k'=1}^{C} exp(\omega_{k'0} + \omega_{k'}^T x_n)} x_n \tag{36}$$

Sum over all ω_k:

$$\lambda \sum_{k=1}^{C} \omega_k = \sum_{n=1}^{N} x_n - \sum_{n=1}^{N} x_n = 0 \tag{37}$$

Thus, it implies that $\sum_{k=1}^{C} \omega_{k,j} = 0$ for all $j = 1 : D$. Proof done.

■

4.1 Practical Logistic Regression on Real Data

(4.1.*b*) For batch Gradient Descent, following parameters are chosen:

$$\begin{aligned} \textit{For raw data}: \ \textit{step size} &= 0.00002 \quad \textit{iterations} = 10^6 \\ \textit{For standardized data}: \ \textit{step size} &= 10 \qquad\quad \textit{iterations} = 200 \end{aligned}$$

In the real implementation, step size would be further divided by m (i.e. the number of samples). Following are the evolution of training accuracies as a function of training iterations.

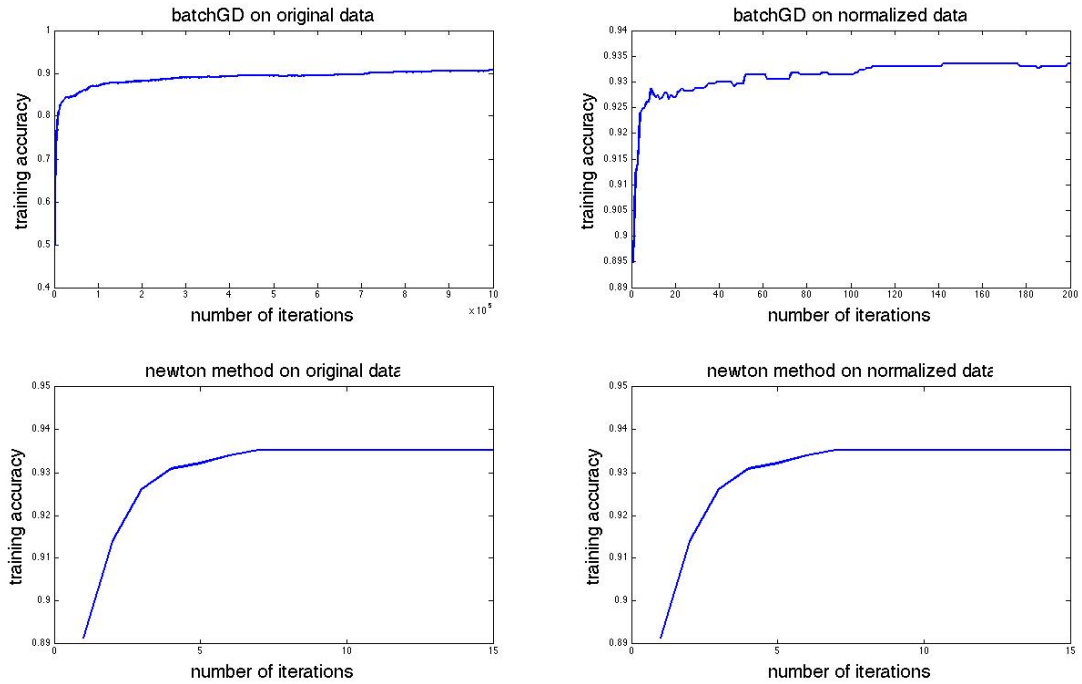

Figure 1: Training Accuracy vs. Training iterations

Training accuracies for raw and standardized data are as follows:

Accuracy	raw	stadardized
batchGD	0.813559	0.932638
Newton	0.934376	0.934376
glmfit	0.934376	0.934376

Table 1: training accuracies

Testing accuracies for raw and standardized data are as follows:

Accuracy	raw	stadardized
batchGD	0.806522	0.920000
Newton	0.922609	0.917826
glmfit	0.922609	0.900870

Table 2: testing accuracies

Notice that the accuracies would change if try it many times, since every time we randomly shuffle the original data and then split them into two parts. Each time training dataset would be different, so as testing dataset. But the basic rule is that on raw data, Newton's method and glmfit perfom similar, and better than batchGD. Since batchGD uses same step size for all features while features can have different deviations, thus making different impact on the error function. For the standardized data, all methods can perform well on the training samples, but batchGD usually perform bettern than Newton and glmfit.

For the batchGD accuracy given in the tables, I used iteration number as 10^6 to make it similar to what we get from glmfit function. The default value of iteration number for batchGD is 10^4, the accuracy would be around 80%.

From Figure 1 and Table 1 & 2 we can see that to get similar accuracy as glmfit, the **iteration number needed** are:

iterations	raw	stadardized
batchGD	1000000	50
Newton	10	10

Table 3: iteration numbers needed

(**4.1.*c***) Figure for MI-vs-FeatureID and PCC-vs-FeatureID:

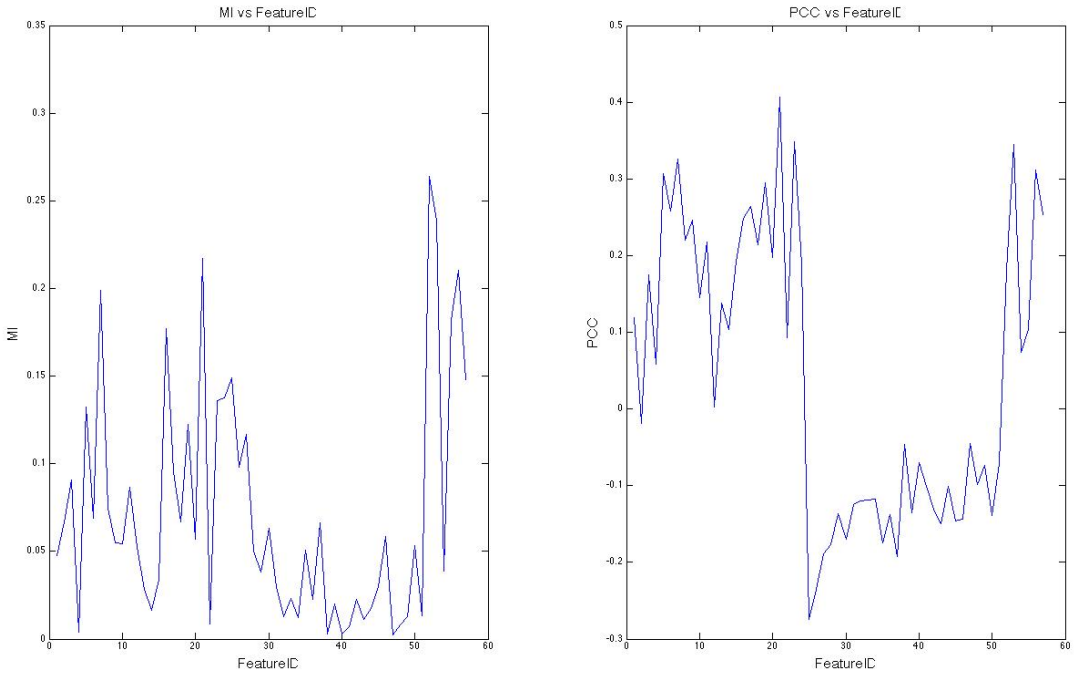

Figure 2: MI-vs-FeatureID & PCC-vs-FeatureID

IDs of chosen 20 features of highest mutual information values, in decreasing order of MI: $52 > 53 > 21 > 56 > 7 > 55 > 16 > 25 > 57 > 24 > 23 > 5 > 19 > 27 > 26 > 17 > 3 > 11 > 8 > 6$.

After training logistic regression on the above 20 features, we get the accuracy as:

$$\begin{aligned} \textit{Training Accuracy} &= 0.906128 \\ \textit{Testing Accuracy} &= 0.893913 \end{aligned}$$

Three features with lowest PCC

Among these 20 features, the lowest 3 features is No. 8, 15, and 14, i.e. their initial IDs are 25, 27, and 26.

Discretize chosen feature

We then discretize these 3 features using 10 equal width bins, i.e. each of these 3 features will thus become binarized 10 features. Training and testing data should be decretized together here and then split them apart according to their original ordering.

Accuracy after discretizing

After training logistic regression on the discretized features together with other normalized 17 features, we get the following accuracy:

$$\begin{aligned} Training\ Accuracy &= 0.894394 \\ Testing\ Accuracy &= 0.893913 \end{aligned}$$

Please note that the accuracies and ordering of 20 features may differ in different runnings. But the lowest PCC feature remain always the same (with ID 25, 27, and 26). After discretizing these thress features using my discretization method, usually training and testing accuracy would decrease. One possible explanation is that features with low PCC has weak relationship with the label, by discretizing it into many more features, we are basically enlarging its influence on the learning parameter, which will possibly lead prediction away from the true value.

■

4.2 Generative Model and Discriminative Model

(4.2.*a*) The parameters for model1 are as the follows:

$$\pi_1 = 0.285714, \qquad \pi_2 = 0.285714, \qquad \pi_3 = 0.428571$$

$$\mu_1 = \begin{bmatrix} 1.581962 & 0.018605 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} -1.529947 & 1.526382 \end{bmatrix}, \quad \mu_3 = \begin{bmatrix} -1.494856 & -1.446381 \end{bmatrix}$$

$$S1 = \begin{bmatrix} 0.974966 & 0 \\ 0 & 2.014032 \end{bmatrix}, \quad S2 = \begin{bmatrix} 1.033020 & 0 \\ 0 & 1.031024 \end{bmatrix}, \quad S3 = \begin{bmatrix} 1.019465 & 0 \\ 0 & 1.035343 \end{bmatrix}$$

Training and testing accuracy:

$$\begin{aligned} Training\ Accuracy &= 0.903429 \\ Testing\ Accuracy &= 0.898743 \end{aligned}$$

(4.2.*b*) The parameters for model2 are as the follows:

$$\pi_1 = 0.285714, \qquad \pi_2 = 0.285714, \qquad \pi_3 = 0.428571$$

$$\mu_1 = \begin{bmatrix} 1.581962 & 0.018605 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} -1.529947 & 1.526382 \end{bmatrix}, \quad \mu_3 = \begin{bmatrix} -1.494856 & -1.446381 \end{bmatrix}$$

$$S1 = \begin{bmatrix} 1.723878 & 0 \\ 0 & 1.857899 \end{bmatrix}, \quad S2 = \begin{bmatrix} 1.723878 & 0 \\ 0 & 1.857899 \end{bmatrix}, \quad S3 = \begin{bmatrix} 1.723878 & 0 \\ 0 & 1.857899 \end{bmatrix}$$

Training and testing accuracy:

$$\begin{aligned} Training\ Accuracy &= 0.869143 \\ Testing\ Accuracy &= 0.860800 \end{aligned}$$

(**4.2.*c***) The parameters for model3 are as the follows:

$$\beta_1 = \begin{bmatrix} -0.153584 & 3.259004 & 0.725945 \end{bmatrix}$$
$$\beta_2 = \begin{bmatrix} -0.699541 & -0.212990 & 1.821362 \end{bmatrix}$$
$$\beta_3 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

Training and testing accuracy:

$$\begin{aligned} \textit{Training Accuracy} &= 0.895143 \\ \textit{Testing Accuracy} &= 0.884571 \end{aligned}$$

(**4.2.*d***) The contour of distribution is as follows:

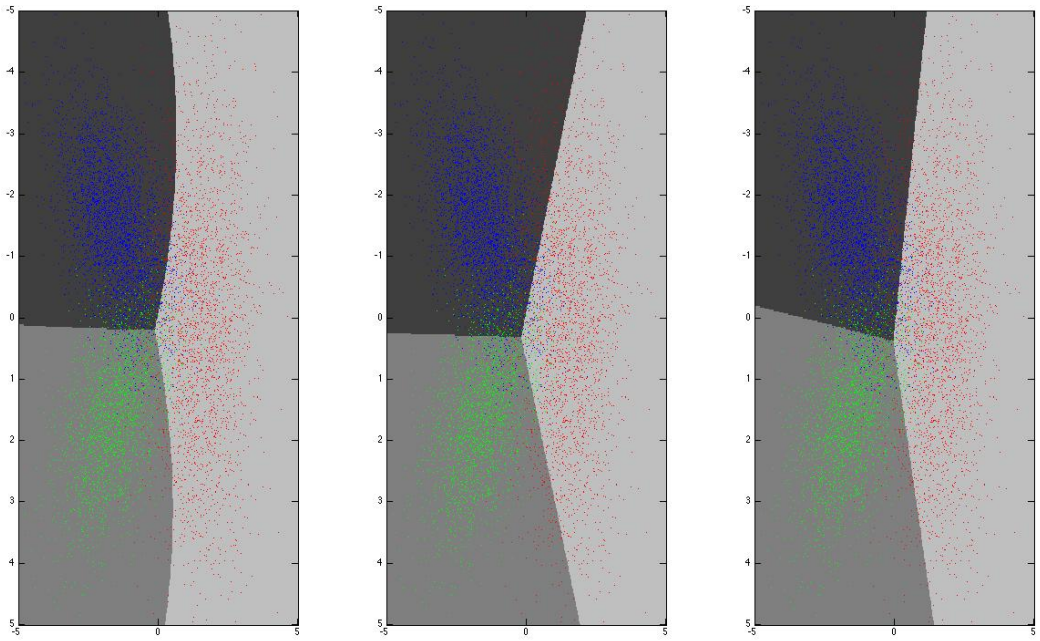

Figure 3: contour of distribution

Analysis:

For model2 and model3, the borders would be linear, while the border for model1 is non-linear. Let's take a look at the border between calss 1 and class 2, i.e. $p(y = 1 \mid x) = p(y = 2 \mid x)$.

For model1 and model2, the border would be:

$$\frac{\pi_1 exp[-\frac{1}{2}(x-\mu_1)^T S_1^{-1}(x-\mu_1)]}{(2\pi)^{D/2}{S_1}^{1/2}} = \frac{\pi_2 exp[-\frac{1}{2}(x-\mu_2)^T S_2^{-1}(x-\mu_2)]}{(2\pi)^{D/2}{S_2}^{1/2}} \tag{38}$$

According to the assumption in model2, $S_1 = S_2$, thus equation (38) would become:

$$\begin{aligned} \pi_1 exp[-\frac{1}{2}(x-\mu_1)^T S^{-1}(x-\mu_1)] &= \pi_2 exp[-\frac{1}{2}(x-\mu_2)^T S^{-1}(x-\mu_2)] \\ \ln\frac{\pi_1}{\pi_2} &= \sum_{i=1}^{D}\frac{(x_i-\mu_{1,i})^2-(x_i-\mu_{2,i})^2}{2S_i} \\ \ln\frac{\pi_1}{\pi_2} &= \sum_{i=1}^{D}\frac{2(\mu_{2,i}-\mu_{1,i})x_i+(\mu_{1,i}^2-\mu_{2,i}^2)}{2S_i} \end{aligned} \tag{39}$$

Equation (39) is exactly a linear function of x, thus under the assumption that $S_1 = S_2$, (38) would be a linear function of x, while without that assumption, (38) won't be linear.

For model3, the border would be:

$$\begin{aligned} \frac{exp(\omega_1^T x)}{\sum_{i=1}^{3} exp(\omega_i^T x)} &= \frac{exp(\omega_2^T x)}{\sum_{i=1}^{3} exp(\omega_i^T x)} \\ (\omega_1-\omega_2)^T x &= 0 \end{aligned} \tag{40}$$

Equation 40 is exactly a linear function of x.

Based on the analysis above, we know why the shapes of the borders in model1, model2, and model3 are different, and what shapes would they be. The shapes can be reflected well in the figures.

(**4.2.*e***) Figures of testing accuracy w.r.t training data sizes for three models:

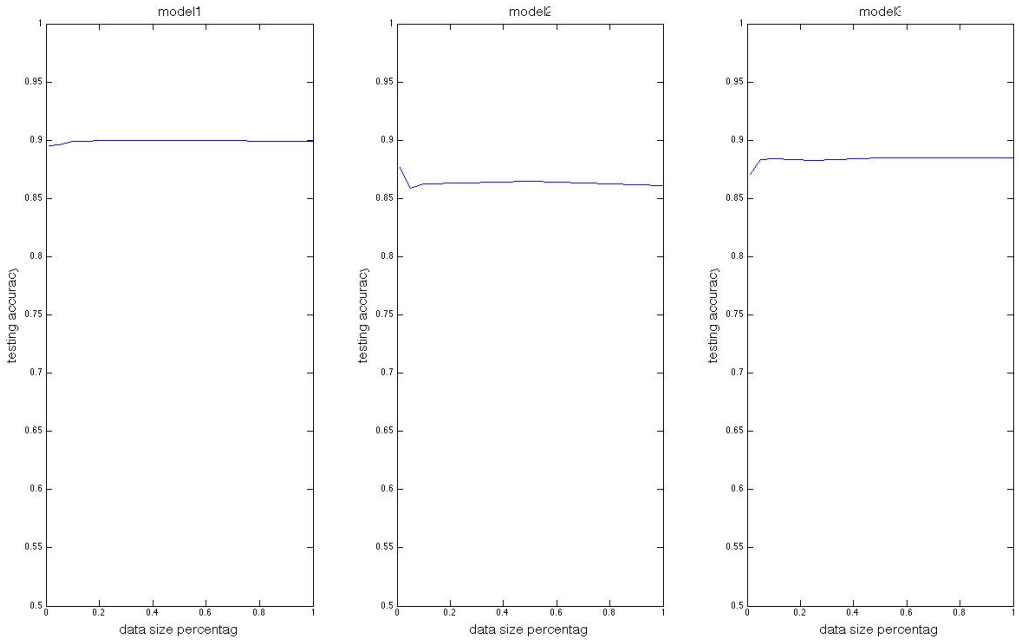

Figure 4: Accuracy w.r.t training data size

Detailed accuracies are as follows:

1. Randomly pick 1% of training data:
   model1: training accuracy = 0.828571, testing accuracy = 0.894971
   model2: training accuracy = 0.800000, testing accuracy = 0.877029
   model3: training accuracy = 0.914286, testing accuracy = 0.870400
2. Randomly pick 5% of training data:
   model1: training accuracy = 0.897143, testing accuracy = 0.896229
   model2: training accuracy = 0.874286, testing accuracy = 0.858857
   model3: training accuracy = 0.897143, testing accuracy = 0.883657
3. Randomly pick 10% of training data:
   model1: training accuracy = 0.900000, testing accuracy = 0.899200
   model2: training accuracy = 0.868571, testing accuracy = 0.862857
   model3: training accuracy = 0.897143, testing accuracy = 0.883886
4. Randomly pick 25% of training data:
   model1: training accuracy = 0.904000, testing accuracy = 0.900114
   model2: training accuracy = 0.867429, testing accuracy = 0.862971
   model3: training accuracy = 0.885714, testing accuracy = 0.882971
5. Randomly pick 50% of training data:
   model1: training accuracy = 0.904000, testing accuracy = 0.899771
   model2: training accuracy = 0.874286, testing accuracy = 0.865029
   model3: training accuracy = 0.897143, testing accuracy = 0.884571
6. Randomly pick 100% of training data:
   model1: training accuracy = 0.903429, testing accuracy = 0.898743
   model2: training accuracy = 0.869143, testing accuracy = 0.860800
   model3: training accuracy = 0.895143, testing accuracy = 0.884571

■

4.3 Practical Logistic Regression on Toy Data

(**4.3.*a*.1**) The **heldout accuracies** for **dataset1** are as follows:

grid size	2	4	8	16
No.1	0.925000	0.900000	1.000000	1.000000
No.2	0.925000	0.925000	1.000000	1.000000
No.3	1.000000	1.000000	1.000000	1.000000
No.4	1.000000	1.000000	0.925000	1.000000
No.5	0.900000	0.900000	0.875000	0.750000
No.6	1.000000	1.000000	0.825000	0.950000
No.7	0.950000	0.950000	0.975000	0.700000
No.8	0.925000	0.925000	0.925000	0.975000
No.9	0.900000	0.900000	0.900000	0.950000
No.10	0.975000	0.975000	0.900000	0.800000
Mean	0.950000	0.947500	0.932500	0.912500

Table 4: heldout accuracies for dataset1

The **heldout accuracies** for **dataset2** are as follows:

grid size	2	4	8	16
No.1	0.050000	0.975000	0.975000	1.000000
No.2	0.025000	0.950000	0.825000	1.000000
No.3	0.525000	0.925000	0.925000	0.950000
No.4	0.000000	0.950000	0.975000	0.925000
No.5	0.000000	0.950000	0.975000	1.000000
No.6	0.650000	0.975000	0.975000	1.000000
No.7	0.525000	1.000000	1.000000	1.000000
No.8	0.700000	0.975000	0.975000	1.000000
No.9	0.000000	0.975000	1.000000	0.925000
No.10	0.575000	0.950000	0.975000	0.800000
Mean	0.305000	0.962500	0.960000	0.960000

Table 5: heldout accuracies for dataset2

The **heldout accuracies** for **dataset3** are as follows:

grid size	2	4	8	16
No.1	0.650000	0.700000	1.000000	1.000000
No.2	0.700000	0.800000	0.900000	0.850000
No.3	0.350000	0.950000	0.950000	1.000000
No.4	0.700000	0.800000	1.000000	1.000000
No.5	0.450000	0.950000	1.000000	1.000000
No.6	0.750000	0.800000	1.000000	0.900000
No.7	0.600000	0.850000	0.800000	0.900000
No.8	0.650000	1.000000	1.000000	0.800000
No.9	0.700000	0.900000	1.000000	1.000000
No.10	0.700000	0.900000	1.000000	1.000000
Mean	0.625000	0.865000	0.965000	0.945000

Table 6: heldout accuracies for dataset3

The **heldout accuracies** for **dataset4** are as follows:

grid size	2	4	8	16
No.1	0.000000	0.825000	0.950000	0.975000
No.2	0.000000	1.000000	0.975000	0.975000
No.3	0.000000	0.850000	0.950000	0.900000
No.4	0.000000	0.850000	0.975000	0.925000
No.5	0.000000	0.875000	0.875000	0.900000
No.6	0.100000	0.700000	0.975000	0.950000
No.7	0.000000	0.650000	0.925000	0.675000
No.8	0.125000	0.750000	0.950000	0.975000
No.9	0.150000	0.750000	1.000000	1.000000
No.10	0.000000	0.675000	1.000000	0.950000
Mean	0.037500	0.792500	0.957500	0.922500

Table 7: heldout accuracies for dataset4

Thus the **training**, **heldout**, and **testing accuracies** for 4 models with **chosen grid size** are:

Statistics	training	heldout	testing	grid
dataset 1	0.950000	0.950000	0.958889	2
dataset 2	0.942500	0.962500	0.923889	4
dataset 3	0.980000	0.965000	0.948333	8
dataset 4	0.957500	0.957500	0.941667	8

Table 8: accuracies and chosen grid size for 4 datasets

(**4.3.*a*.2**) The **heldout accuracies** of **l_2 norm regularization** for **dataset1** are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.925000	0.925000	0.950000	0.950000
No.2	0.900000	0.900000	0.925000	0.925000
No.3	0.925000	0.925000	0.950000	0.925000
No.4	0.875000	0.875000	1.000000	1.000000
No.5	0.925000	0.925000	0.975000	0.975000
No.6	0.975000	0.975000	1.000000	1.000000
No.7	0.975000	0.975000	0.975000	0.950000
No.8	0.875000	0.925000	0.875000	0.925000
No.9	0.975000	0.975000	0.950000	0.975000
No.10	0.925000	0.925000	0.975000	0.925000
Mean	0.927500	0.932500	0.957500	0.955000

Table 9: heldout accuracies of l_2 norm regularization for dataset1

The **l_2 norm regularization accuracies** for **dataset1** are as follows:

lambda	1	0.1	0.01	0.001
training	0.970000	0.970000	0.967500	0.970000
heldout	0.837500	0.837500	0.962500	0.957500
testing	0.945278	0.942500	0.933889	0.942500

Table 10: all accuracies of l_2 norm regularization for dataset1

The **heldout accuracies** of **l_2 norm regularization** for **dataset2** are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.850000	0.850000	1.000000	1.000000
No.2	0.950000	0.950000	0.950000	0.975000
No.3	0.975000	0.975000	0.975000	0.975000
No.4	0.950000	0.950000	0.950000	0.950000
No.5	0.950000	0.950000	0.975000	0.975000
No.6	1.000000	1.000000	1.000000	1.000000
No.7	0.850000	0.850000	1.000000	1.000000
No.8	0.900000	0.900000	0.975000	0.975000
No.9	0.875000	0.875000	0.950000	0.925000
No.10	0.900000	0.900000	0.950000	0.950000
Mean	0.920000	0.920000	0.972500	0.972500

Table 11: heldout accuracies of l_2 norm regularization for dataset2

The **l_2 norm regularization accuracies** for **dataset2** are as follows:

lambda	1	0.1	0.01	0.001
training	0.985000	0.985000	0.985000	0.985000
heldout	0.920000	0.920000	0.972500	0.972500
testing	0.968056	0.968056	0.968056	0.968056

Table 12: all accuracies of l_2 norm regularization for dataset2

The **heldout accuracies** of **l_2 norm regularization** for **dataset3** are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.750000	0.750000	1.000000	1.000000
No.2	1.000000	1.000000	1.000000	1.000000
No.3	0.800000	0.800000	0.950000	0.950000
No.4	0.950000	0.950000	0.950000	1.000000
No.5	0.900000	0.900000	0.900000	0.900000
No.6	0.900000	0.900000	0.950000	0.950000
No.7	0.950000	0.950000	0.950000	0.950000
No.8	0.850000	0.850000	0.950000	0.950000
No.9	0.900000	0.900000	0.900000	0.950000
No.10	0.800000	0.800000	0.850000	0.850000
Mean	0.880000	0.880000	0.940000	0.950000

Table 13: heldout accuracies of l_2 norm regularization for dataset3

The **l_2 norm regularization accuracies** for **dataset3** are as follows:

lambda	1	0.1	0.01	0.001
training	0.980000	0.980000	0.980000	0.980000
heldout	0.880000	0.880000	0.940000	0.950000
testing	0.948333	0.948333	0.948333	0.964444

Table 14: all accuracies of l_2 norm regularization for dataset3

The **heldout accuracies** of $\mathbf{l_2}$ **norm regularization** for **dataset4** are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.700000	0.675000	0.975000	0.975000
No.2	0.725000	0.725000	0.950000	0.950000
No.3	0.950000	0.950000	0.950000	0.950000
No.4	0.800000	0.800000	0.800000	0.925000
No.5	0.975000	0.975000	0.950000	0.975000
No.6	0.975000	0.975000	0.950000	0.975000
No.7	1.000000	1.000000	0.950000	1.000000
No.8	0.750000	0.750000	1.000000	1.000000
No.9	0.825000	0.825000	0.825000	0.875000
No.10	0.800000	0.850000	0.950000	0.950000
Mean	0.850000	0.852500	0.930000	0.957500

Table 15: heldout accuracies of l_2 norm regularization for dataset4

The $\mathbf{l_2}$ **norm regularization accuracies** for **dataset4** are as follows:

lambda	1	0.1	0.01	0.001
training	0.957500	0.957500	0.957500	0.957500
heldout	0.850000	0.852500	0.930000	0.957500
testing	0.941667	0.941667	0.941667	0.941667

Table 16: all accuracies of l_2 norm regularization for dataset4

Based on **mean-healdout accuracies**, the accuracies for these four datasets with **chosen lambda** are:

Statistics	training	heldout	testing	lambda
dataset 1	0.967500	0.957500	0.933889	0.01
dataset 2	0.985000	0.972500	0.968056	0.001
dataset 3	0.980000	0.950000	0.964444	0.001
dataset 4	0.957500	0.957500	0.941667	0.001

Table 17: accuracies and chosen lambda for 4 datasets

(**4.3.*a*.3**) For dataset2 and dataset3, the parameters' 3-D bar plots are as follows:

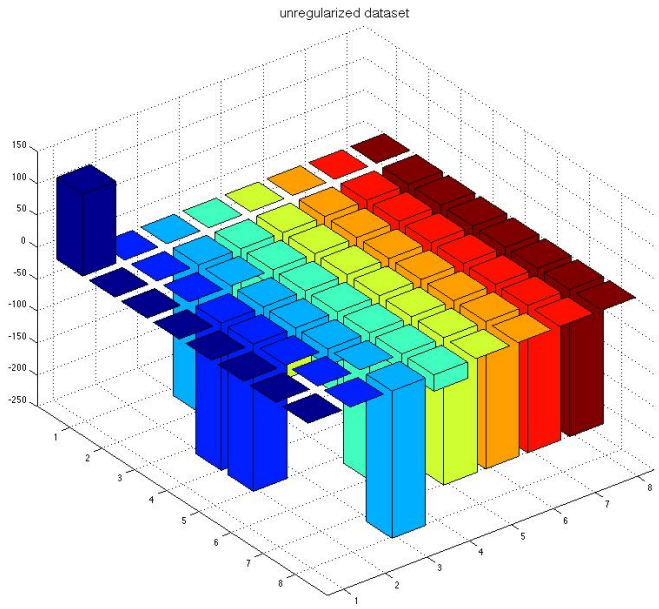

Figure 5: unregularized parameters for dataset2

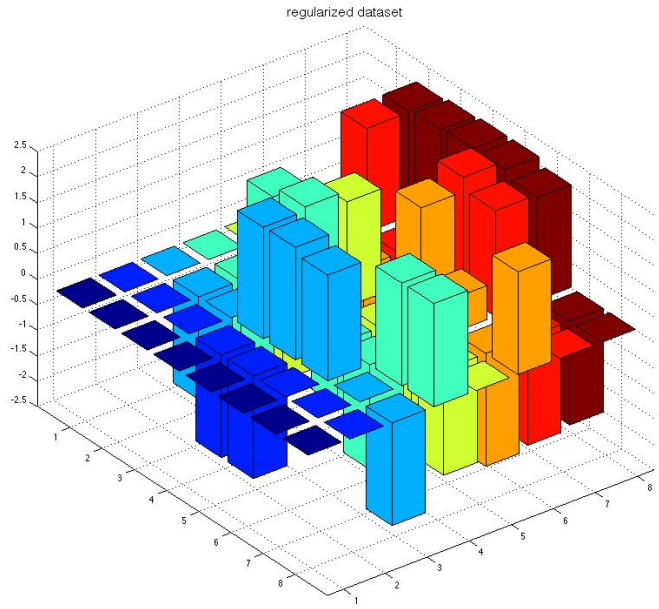

Figure 6: best l_2 regularized parameters for dataset2

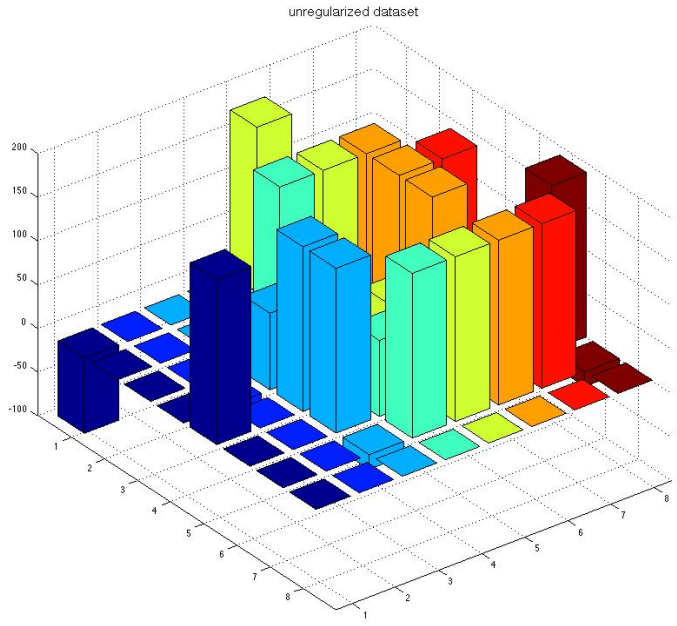

Figure 7: unregularized parameters for dataset3

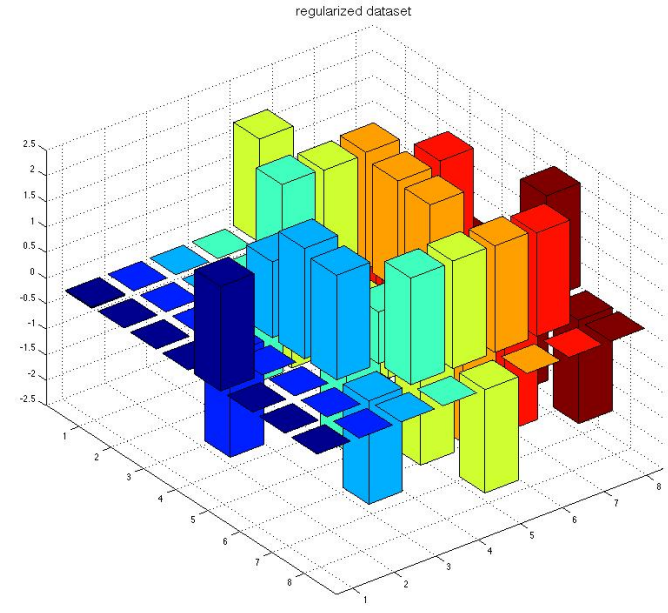

Figure 8: best l_2 regularized parameters for dataset3

Analysis:

From the previous figures, we can see that when **unregularized**, even with the optimal parameter, the absolute value of several dimensions can be very large, with the order of magnitude 10^2. After using the $\mathbf{l_2}$ **regularization**, the absolute value of these dimensions would be significantly reduced, since we add an order of two punishment on every dimension of the parameter (bias parameter not included). And usually, really large or really small parameter reflect overfitting, by adding a reasonable l_2 regularizer, we tend to prevent overfitting.

(4.3.*b*.2) Train logistic regression model (12) with pairwie regularization on these four datasets. The **heldout accuracies** for **dataset1** with pairwise regularization are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.550000	0.950000	0.950000	0.950000
No.2	0.575000	0.900000	0.900000	0.925000
No.3	0.725000	0.975000	0.975000	0.975000
No.4	0.650000	1.000000	1.000000	1.000000
No.5	0.425000	0.975000	0.975000	0.975000
No.6	0.575000	0.975000	0.975000	1.000000
No.7	0.900000	0.950000	0.950000	0.950000
No.8	0.875000	0.875000	0.875000	0.925000
No.9	0.875000	0.950000	0.950000	0.950000
No.10	0.725000	0.950000	0.950000	0.975000
Mean	0.687500	0.950000	0.950000	0.962500

Table 18: heldout accuracies pairwise regularization for dataset1

The **heldout accuracies** for **dataset2** with pairwise regularization are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.425000	0.900000	0.925000	0.950000
No.2	0.975000	0.975000	0.975000	0.950000
No.3	0.875000	0.950000	0.950000	0.975000
No.4	0.450000	0.900000	0.900000	0.925000
No.5	0.450000	0.900000	0.950000	0.950000
No.6	0.975000	0.975000	0.975000	1.000000
No.7	0.325000	0.875000	0.975000	0.975000
No.8	0.400000	0.925000	0.925000	0.950000
No.9	0.400000	0.850000	0.875000	0.875000
No.10	0.800000	0.875000	0.925000	0.950000
Mean	0.607500	0.912500	0.937500	0.950000

Table 19: heldout accuracies pairwise regularization for dataset2

The **heldout accuracies** for **dataset3** with pairwise regularization are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.350000	0.600000	0.900000	0.950000
No.2	0.650000	0.850000	0.900000	1.000000
No.3	0.350000	0.800000	0.900000	1.000000
No.4	0.850000	0.850000	0.850000	0.950000
No.5	0.500000	0.850000	0.850000	0.850000
No.6	0.500000	0.550000	0.600000	0.900000
No.7	0.650000	0.800000	0.850000	0.900000
No.8	0.400000	0.700000	0.750000	0.950000
No.9	0.550000	0.650000	0.900000	0.950000
No.10	0.450000	0.550000	0.750000	0.850000
Mean	0.525000	0.720000	0.825000	0.930000

Table 20: heldout accuracies pairwise regularization for dataset3

The **heldout accuracies** for **dataset4** with pairwise regularization are as follows:

lambda	1	0.1	0.01	0.001
No.1	0.425000	0.925000	0.975000	0.975000
No.2	0.425000	0.925000	0.950000	0.950000
No.3	0.875000	0.875000	0.950000	0.950000
No.4	0.375000	0.800000	0.800000	0.925000
No.5	0.900000	0.900000	0.975000	0.975000
No.6	0.625000	0.900000	0.975000	0.975000
No.7	0.925000	0.950000	1.000000	1.000000
No.8	0.425000	0.925000	1.000000	1.000000
No.9	0.400000	0.775000	0.875000	0.875000
No.10	0.650000	0.925000	0.950000	0.950000
Mean	0.602500	0.890000	0.945000	0.957500

Table 21: heldout accuracies pairwise regularization for dataset4

According to the heldout accuracies of 4 datasets, we choose the best λ and thus get the following **training**, **heldout**, and **testing** accuracies:

Statistics	training	heldout	testing	lambda
dataset 1	0.965000	0.962500	0.953056	0.001
dataset 2	0.970000	0.950000	0.947778	0.001
dataset 3	0.965000	0.930000	0.931111	0.001
dataset 4	0.957500	0.957500	0.941667	0.001

Table 22: accuracies and chosen lambda for 4 datasets with pairwise regularization

(**4.3.*b*.3**) For dataset2 and dataset3, the parameters' 3-D bar plots of model (11) and model (12), or the unregularized one and the pairwise regularized one as in the problem statement, are as follows:

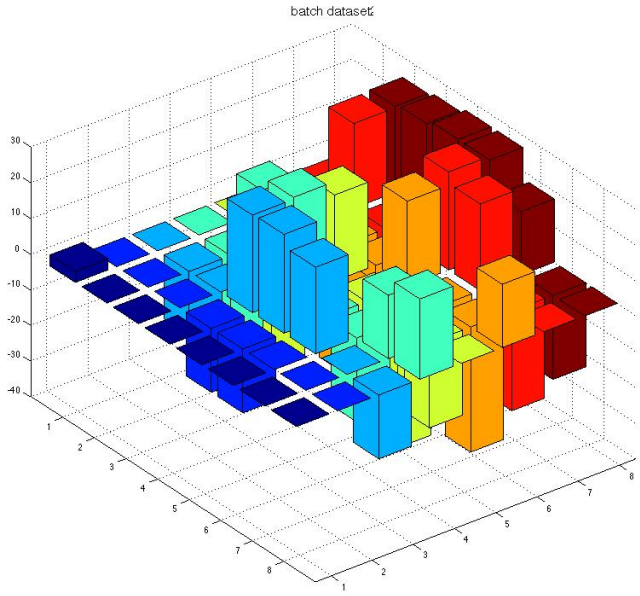

Figure 9: parameters for dataset2 of model (11)

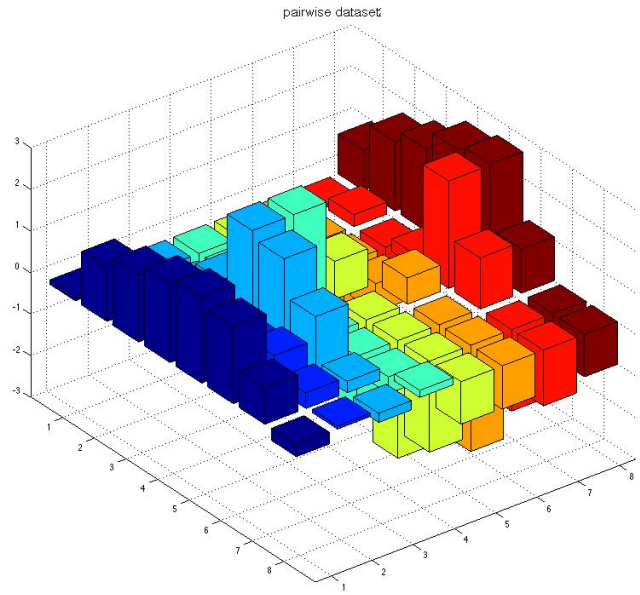

Figure 10: parameters for dataset2 of model (12)

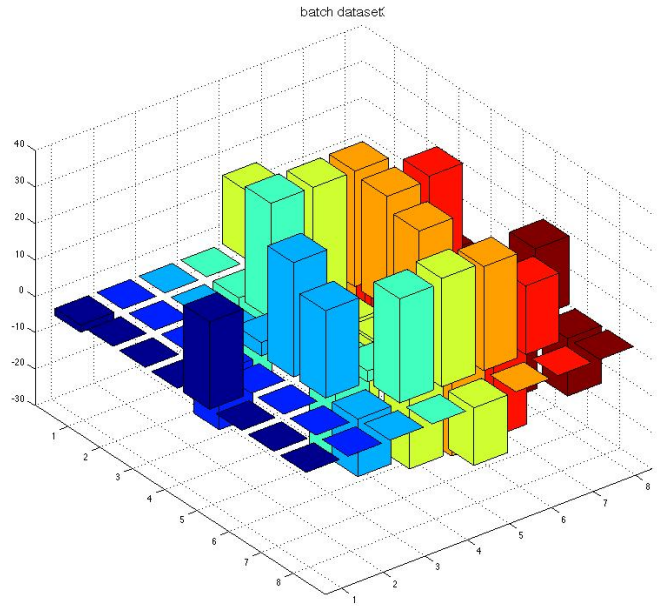

Figure 11: parameters for dataset3 of model (11)

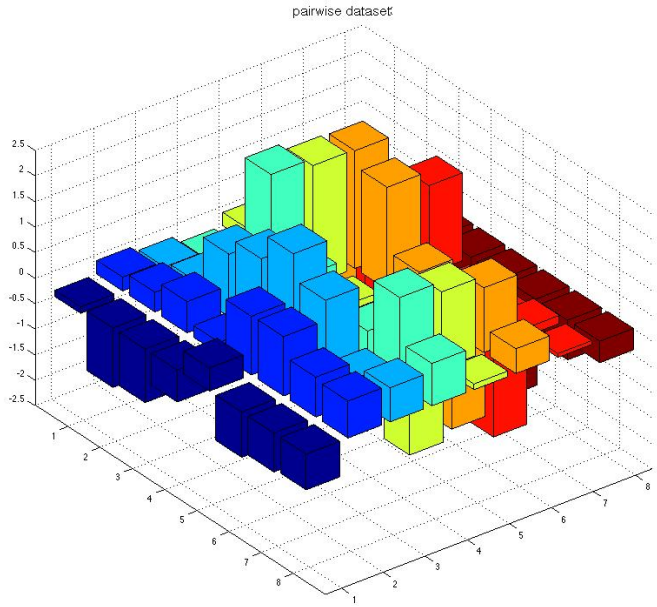

Figure 12: parameters for dataset3 of model (12)

Analysis:

From the previous four figures, we can see that when **unregularized**, neighbor dimensions in the parameter can very largely. By adding **piarwise regularizer**, we are punishing neighboring dimensions with differences, thus reducing the variance between neighboring dimensions of parameter, and this can be reflected from these figures. This regularizer can be useful if we are sure to some extent that neighbor features contribute similarly to the label.

■