CSCI 5525 Machine Learning HW1

Problem3

Questions

(a)

No the data are not well-modeled by a linear discriminant after being projected on $\mathbb{R}$. In the Fig. 1, we can see that the projected data is not linearly separable and there are around 30% overlaps between two classes. Then we cannot expect a high accuracy with a linear discriminant.

(b)

No we cannot project the Boston50 data to $\mathbb{R}^2$. Since for a data with K classes, the between-class covariance matrix S_B has rank at most equal to $(K-1)$ and so there are at most $(K-1)$ nonzero eigenvalues. And there are thus at most $(K-1)$ eigenvectors to span the subspace. Boston data is a 2-class data so we can only project it to $\mathbb{R}$.

LDA1dProjection

The weight vector w that maximize the ratio of the between-class variance to the within-class variance is given by

$$\mathbf{w} \propto \mathbf{S}_\mathbf{w}^{-1}(\mathbf{m1} - \mathbf{m2}) \tag{1}$$

where $\mathbf{m1}$ and $\mathbf{m2}$ are the mean of the two classes respectively. Fig.1 is the histogram the projected points.

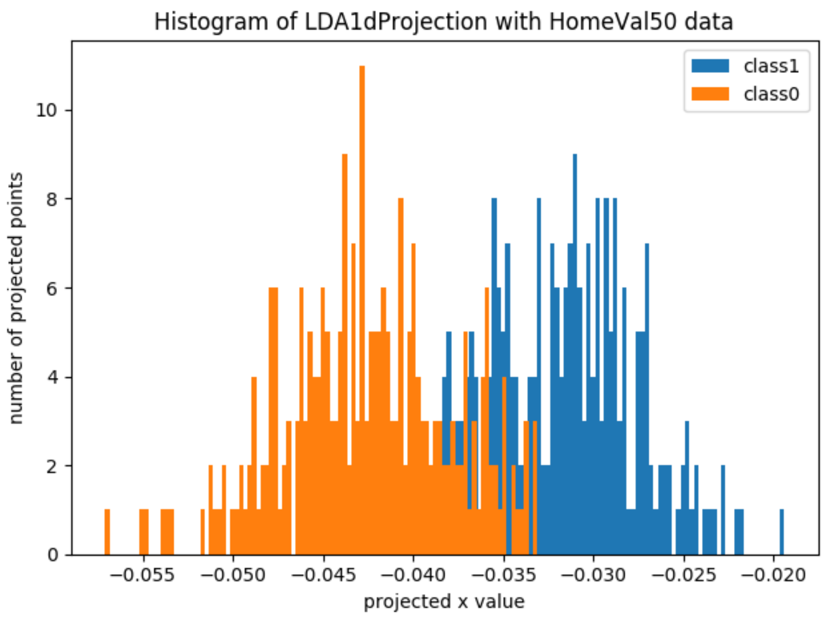


Figure 1: Histogram the projected points

LDA2dGaussGM

The projection matrix in Fisher's discrimiant for multiple classes is derived by maximizing the quantity

$$J(\mathbf{W}) = \mathrm{Tr}\left\{(\mathbf{W}\mathbf{S}_\mathbf{W}\mathbf{W}^T)^{-1}(\mathbf{W}\mathbf{S}_\mathbf{B}\mathbf{W}^T)\right\} \tag{2}$$

and the solution is the concatenation of eigenvectors of $\mathbf{S}_\mathbf{W}^{-1}\mathbf{S}_\mathbf{B}$ that correspond to the D' (the dimension of the subspace here is 2) largest eigenvalues. Note that $\mathbf{W}$ may be singular due to numerical issues or the low-covariance of some input data features. And we can choose to process the data first or add add a small $\alpha\mathbf{I}$ to it.

Then we formulate the bi-variate Gaussian generative modeling to do 10-class classification by by estimating π_k for class priors $P(C_k)$ and parameters of Gauss distribution $\boldsymbol{\mu}_k$, $\mathbf{C}_k$ for likelihood $P(\mathbf{x}|C_k)$ for each class k

$$\pi_k = \frac{N_k}{N} \tag{3}$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n\in C_k}\mathbf{x}_n \tag{4}$$

$$\mathbf{C}_k = \frac{1}{N_k}\sum_{n\in C_k}(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \tag{5}$$

From the Bayes' rule, we make the prediction based on the prediction score $log(p(\mathbf{x}|C_k))+log(p(C_k))$, i.e., compute the scores for all k and classify the data point to the class with the largest prediction score.

We train and evaluate the model using digits data. The train error rates, test error rates and test error standard deviation are in Table 1. The error rate is rather high and it might because projecting on $\mathbb{R}^2$ is not enough to achieve high accuracy. We tried to project the data to higher dimension (e.g., $\mathbb{R}^5$) and the accuracy can be over 90%.

Table 1: LDA2dGaussGM statics

	fold0	fold1	fold2	fold3	fold4	fold5	fold6	fold7	fold8	fold9
train error mean	28.3%	29.5%	28.1%	30.2%	29.5%	29.8%	29.9%	28.3%	28.2%	28.8%
test error mean	31.1%	33.6%	31.8%	33.3%	32.7%	32.7 %	32.3%	31.4%	30.3%	32.8%
test error std	0.03	0.04	0.04	0.04	0.02	0.04	0.05	0.03	0.02	0.03

Problem4

Logistic Regression

In multiclass logistic regression, we make the prediction based on the posterior probabilities

$$P(C_k|\mathbf{x}) = y_k(\mathbf{x}) = \frac{\exp{(a_k)}}{\sum_j \exp{(a_j)}} \tag{6}$$

where a_k is the "activation" for class k

$$a_k = \mathbf{w}_k^T\mathbf{x} \tag{7}$$

so the task is to learn the weight vector $\mathbf{w}_k$ for every class and we use newton method (or iterative reweighted least squares) to optimize the cross-entropy loss function l and the corresponding gradient and hessian w.r.t. the concatenated weight vector $\mathbf{w} = [\mathbf{w}_1^T, \ldots, \mathbf{w}_K^T]^T$ are

$$\nabla_j = \nabla_{\mathbf{w}_j} l = \sum_{n=1}^{N} (y_{nj} - t_{nj})\mathbf{x}_n \tag{8}$$

$$\mathbf{H}_{ij} = \nabla_{\mathbf{w}_i} \nabla_{\mathbf{w}_j} l = \sum_{n=1}^{N} y_{ni}(\mathbf{I}_{ij} - y_{nj})\mathbf{x}_n \mathbf{x}_n^T \tag{9}$$

The the iterative update would be

$$\mathbf{w} = \mathbf{w} - \lambda \mathbf{H}^{-1}(\mathbf{w})\nabla(\mathbf{w}) \tag{10}$$

where λ is the learning rate. Note that $\mathbf{H}$ may be not positive semidefinite due to numerical issue, and we can add a small $\alpha\mathbf{I}$ to it to do the hessian modification.

Naive Bayes with Marginal Gaussian distributions

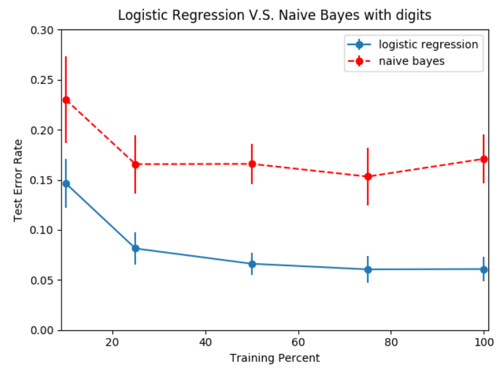
Naive Bayes assumes features are independent with each other, which means

$$p(x_1, \ldots, x_p | C_k) = \prod_{i=1}^{p} p(x_i | C_k) \tag{11}$$

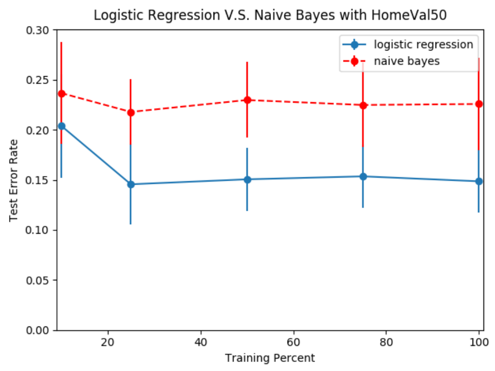
For each $p(x_i|C_k)$ we use a Gaussian distribution with parameters μ_{ik}, σ_{ik} to model it. The Gaussian parameters and priors for each class can be easily estimated from the data.

From the Bayes' rule, we make the prediction based on the prediction score $log(p(\mathbf{x}|C_k)) + log(p(C_k))$, i.e., compute the scores for all k and classify the data point to the class with the largest prediction score.

We train and evaluate the models on the digits data and HomeVal50 respectively. Fig.2 shows the learning curves of logistic regression and naive Bayes where the error bar size is one standard deviation. There are several conclusions from the figure: (1) The two classifiers tend to perform better when the training percentage increases (2) logistic regression always performs better than naive Bayes.



(a) With digits data



(b) With HomeVal50 data

Figure 2: The learning curves of logistic regression and naive Bayes

1. (a) The expected loss is

$$E[L(f(x),y)] = \int_x \{ \int_y L(f(x),y)\, p(y|x)\, dy \}\, p(x) dx$$

Since for every x, the value of $f(x)$ can be independently chosen, so

$$\min_{f(x)} E_{(x,y)}[L(f(x),y)] \Leftrightarrow \min_{f(x)} \int_y L(f(x),y)\, P(y|x)\, dy$$

When $L(f(x),y) = (f(x)-y)^2$, we take the derivative w.r.t $f(x)$ and set it to 0

$$\frac{\partial}{\partial f(x)} \int_y (f(x)-y)^2 P(y|x) dy$$

$$= 2\int_y (f(x)-y) P(y|x) dy$$

$$\int_y (f(x)-y) P(y|x) dy = 0$$

$$\Leftrightarrow \int_y f(x) P(y|x) dy = \int_y y\, P(y|x) dy$$

$$\Leftrightarrow f(x) = E_y[y|x]$$

(b) When $L(f(x),y) = |f(x)-y|$, we take the derivative w.r.t $f(x)$ and set it to 0

$$\frac{\partial}{\partial f(x)} \int_y |f(x)-y|\, P(y|x) dy$$

$$= \frac{\partial}{\partial f(x)} \int_{-\infty}^{f(x)} (f(x)-y) P(y|x) dy + \frac{\partial}{\partial f(x)} \int_{f(x)}^{\infty} (y-f(x)) P(y|x) dy$$

$$= \int_{-\infty}^{f(x)} \frac{\partial (f(x)-y)}{\partial f(x)} P(y|x) dy + (f(x)-f(x)) \frac{\partial f(x)}{\partial f(x)} - (f(x)-f(x)) \frac{\partial (-\infty)}{\partial f(x)}$$

$$+ \int_{f(x)}^{\infty} \frac{\partial (y-f(x))}{\partial f(x)} P(y|x) dy + (f(x)-f(x)) \frac{\partial \infty}{\partial f(x)} - (f(x)-f(x)) \frac{\partial f(x)}{\partial f(x)}$$

$$= \int_{-\infty}^{f(x)} P(y|x) dy - \int_{f(x)}^{\infty} P(y|x) dy$$

where we use Leibniz's Rule in taking the derivative, then set it to 0

$$\int_{-\infty}^{f(x)} P(y|x) dy - \int_{f(x)}^{\infty} P(y|x) dy = 0$$

$$\Leftrightarrow \int_{-\infty}^{f(x)} P(y|x) dy = \int_{f(x)}^{\infty} P(y|x) dy$$

$\Leftrightarrow$ $f(x)$ is the conditional median of y distribution

2. We first find the optimal Bayes classifier by minimizing the total error rate.
For the classifier f, the error rate on the (x, y) domain D is

$$L(f) = P(f(x) \neq y)$$
$$= E_{(x,y)\sim D}[f(x) \neq y]$$
$$= E_{(x,y)\sim D}[1(f(x) \neq y)]$$
$$= \int_x \sum_y 1(f(x) \neq y) P(y|x) P(x) dx \qquad // \ 1 \text{ is the indicator function}$$

$$\min_f \int_x \sum_y 1(f(x) \neq y) P(y|x) P(x) dx$$
$$\Leftrightarrow \min_f \sum_y 1(f(x) \neq y) P(y|x) \qquad // \ f(x) \text{ can be independently chosen of } x$$
$$\Leftrightarrow \min_f \sum_{j=1}^{M} 1(f(x) \neq C_j) P(C_j|x)$$

Among all $P(C_j|x)$, we should pick $f(x)$ to make the term with largest $P(C_j|x)$ zero, thus we can minimize the expectation. So the optimal classifier is

$$f^*(x) = \underset{C_j}{\operatorname{argmax}} P(C_j|x)$$

Then the error rate of $f^*(x)$ for class $y = C_j$ is

$$err[y = C_j] = \int_x 1(f^*(x) \neq C_j) P(C_j|x) P(x) dx$$
$$= \int_{x \notin R_j} P(C_j|x) P(x) dx$$
$$= \sum_{i=1}^{M} \int_{x \in R_i} P(C_j|x) P(x) dx - \int_{x \in R_j} P(C_j|x) P(x) dx$$