Problem 1.

Q(1)

$E[L] = \int_x \int_y l(f(x), y)p(x, y)dydx$

$E[L] = \int_x \int_y (f(x) - y)^2 p(x, y)dydx$

$E[L] = \int_y \int_x (f(x) - y)^2 p(x, y)dxdy$

If we proceed the calculus of variation to minimize this equation:

$\frac{\delta E[L]}{\delta f(x)} = 2\int (f(x) - y)p(x, y)dy = 0$

f(x) = $\frac{\int yp(x,y)dy}{p(x)} = \int yp(y|x)dy$

Optimal f(x) = $E_y[y|x]$

Q(2)

$\int_x \{\int_y l(f(x), y)p(y|x)dy\}p(x)dx$

For every x, f(x) would be independently chosen. So we do not need to worry about the second integral($\int_x \dots p(x)dx$)

For the absolute, we change it to signum function, the definition of signum function is following:

sgn s = -1, when $s < 0$; sgn s = 0, when $s = 0$; sgn s = 1, when $s > 0$

We can convert our function to: $\int_{f(x)}^{\infty}(f(x) - y)p(y|x)dy + \int_{-\infty}^{f(x)} -(f(x) - y)p(y|x)dy$,then,

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

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

If we take the derivative of this equation and let it be zero:

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

$\therefore$ The optimal f(x) is conditional median of y given x

Problem 2.

$||Aw - b + f||_2^2 + g$

$=||Aw - b||_2^2 + ||f||_2^2 + 2f^T(Aw - b) + g$

$=||Aw - b||_2^2 + ||f||_2^2 + 2f^T Aw - 2f^T b + g$

$\because$ The dimension of c^T is $1*n$ and the dimension of $2f^T A$ is $1*m*m*n = 1*n$.

$\therefore$ we can let $c^T = 2f^T A$.

$\because$ The dimension of d is $1 * 1$ and the dimension of $||f||_2^2 + g - 2f^T b$ is $1 * 1 + 1 * 1 + 1 * m * m * 1 = 1$.

$\therefore$ we can let $||f||_2^2 + g - 2f^T b = d$.

$\therefore ||Aw - b||_2^2 + c^T w + d = ||Aw - b + f||_2^2 + g$

Solve the generalized least squares:

$||Aw - (b - f)||_2^2 = ||Aw - b + f||_2^2$

$=||Aw - b||_2^2 + ||f||_2^2 + 2f^T(Aw - b)$

$\because$ For finding the minimum, $\frac{d(||Aw-b||_2^2 + ||f||_2^2 + 2f^T(Aw-b))}{dw} = 0$

$\therefore 2A^T(Aw - b) = -2A^T f$

$A^T Aw - A^T b = -A^T f$

$A^T Aw = A^T b - A^T f$

$A^T Aw = A^T(b - f)$

$w = (A^T A)^{-1} A^T (b - f)$

Problem 3

Q(1)

Algorithm:

We use Fisher's linear discriminant analysis here. In LDA, our basic idea is to select a direction and project the test sample along this direction. The projected data needs to meet two conditions:

1. Test samples within the same category are close to each other.

2. The distance between different categories is large.

The first is projection. We assume that the original data is a vector, then the projection along the direction is a scalar:

$z = w^T \cdot x (= |w| \cdot |x| \cos\theta)$

For the first point, the samples within the same class are closer. We assume that the number of test samples belonging to the two classes are sums N_1 and N_2. Then we use the variance matrix to characterize the overall distribution within each class:

$$\begin{aligned} C_1 : Var_z[C_1] &= \frac{1}{N_1}\sum_{i=1}^{N_1}(z_i - \overline{z_{c1}})(z_i - \overline{z_{c1}})^T \\ &= \frac{1}{N_1}\sum_{i=1}^{N_1}(w^T x_i - \frac{1}{N_1}\sum_{j=1}^{N_1} w^T x_j)(w^T x_i - \frac{1}{N_1}\sum_{j=1}^{N_1} w^T x_j)^T \\ &= w^T \frac{1}{N_1}\sum_{i=1}^{N_1}(x_i - \overline{x_{c1}})(x_i - \overline{x_{c1}})^T w \\ &= w^T S_1 w \qquad (1) \\ C_2 : Var_z[C_2] &= \frac{1}{N_2}\sum_{i=1}^{N_2}(z_i - \overline{z_{c2}})(z_i - \overline{z_{c2}})^T \\ &= w^T S_2 w \qquad (2) \end{aligned}$$

So the within class distance could be written as: $w^T(S_1 + S_2)w$

For the second point, we can use the mean of the two categories to express this distance:

$$\begin{aligned} (\overline{z_{c1}} - \overline{z_{c2}})^2 &= (\frac{1}{N_1}\sum_{i=1}^{N_1} w^T x_i - \frac{1}{N_2}\sum_{i=1}^{N_2} w^T x_i)^2 \\ &= (w^T(\overline{x_{c1}} - \overline{x_{c2}}))^2 \\ &= w^T(\overline{x_{c1}} - \overline{x_{c2}})(\overline{x_{c1}} - \overline{x_{c2}})^T w \qquad (3) \end{aligned}$$

Combining these two points, since the co-variance is a matrix, we divide these two values to get our loss function and maximize this value:

$$\begin{aligned}\hat{w} = \underset{w}{argmax}\, J(w) &= \underset{w}{argmax}\, \frac{(\overline{z_{c1}} - \overline{z_{c2}})^2}{Var_z[C_1] + Var_z[C_2]} \\ &= \underset{w}{argmax}\, \frac{w^T(\overline{x_{c1}} - \overline{x_{c2}})(\overline{x_{c1}} - \overline{x_{c2}})^T w}{w^T(S_1 + S_2)w} \\ &= \underset{w}{argmax}\, \frac{w^T S_b w}{w^T S_w w} \end{aligned} \tag{4}$$

In this way, we combine the loss function with the original data set and parameters. The following is the partial derivative of this loss function. Note that we actually do not have any requirements for the absolute value of$w$. We only need the direction, so only one equation can be solved:

$$\begin{aligned}&\frac{\partial}{\partial w} J(w) = 2S_b w(w^T S_w w)^{-1} - 2w^T S_b w(w^T S_w w)^{-2} S_w w = 0 \\ &\implies S_b w(w^T S_w w) = (w^T S_b w) S_w w \\ &\implies w \propto S_w^{-1} S_b w = S_w^{-1}(\overline{x_{c1}} - \overline{x_{c2}})(\overline{x_{c1}} - \overline{x_{c2}})^T w \propto S_w^{-1}(\overline{x_{c1}} - \overline{x_{c2}}) \end{aligned} \tag{5}$$

So $S_w^{-1}(\overline{x_{c1}} - \overline{x_{c2}})$ is the direction we need.

From PRML equation (4.34), we also know the expression for the bias in the form: $w_0 = -w^T m$.

Result:

From this problem, we find the w_0 would be 0.03819. The threshold is the -w_0, which is -0.03819.

The training set error rates from 10-fold cross validation would be:

[0.1912087912087912, 0.18681318681318682, 0.22197802197802197, 0.16923076923076918, 0.17142857142857137, 0.1912087912087912, 0.18021978021978025, 0.16263736263736261, 0.22857142857142854, 0.18082788671023964]

The standard deviation is 0.021610124809659348

The training set error rates from 10-fold cross validation would be:

[0.1568627450980392, 0.3529411764705882, 0.13725490196078427, 0.17647058823529416, 0.27450980392156865, 0.019607843137254943, 0.3529411764705882, 0.27450980392156865, 0.019607843137254943, 0.2978723404255319]

The standard deviation is 0.12405504339545924

Q(2)

Algorithm:

Now we consider the generalization of the LDA to K ¿2 classes:

In this part, our within-class covariance matrix to the case of K classes is(from PRML equations 4.40 4.41 4.42):

$S_w = \sum_{k=1}^{K} S_k$

where

$S_k = \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T$

$m_k = \frac{1}{N_k} \sum_{n \in C_k} x_n$

The N_k is the number of class that need to be identified, which is 10 in this problem.

The generalization of the between-class covariance matrix would be:

$S_T = \sum_{n=1}^{N} (x_n - m)(x_n - m)^T$

where

$m = \frac{1}{N} \sum_{n=1}^{N} x_n$

For multidimensions LDA:

$J(w) = Tr(s_W^{-1} s_B) = \prod_{i=1}^{diagonal} \frac{w_i^T S_b w_i}{w_i^T S_w w_i}$, which is generalized Rayleigh quotient. The maximum value is the largest eigenvalue of the matrix $S_w^{-1} S_b$, and the product of the largest d values is the product of the largest d eigenvalues of the matrix $S_w^{-1} S_b$. At this time, the corresponding matrix W is a matrix

formed by the eigenvectors corresponding to the largest d eigenvalues.

For the two dimension LDA, we need to find the matrix W, which is formed by the the eigenvectors corresponding to the largest and the second largest eigenvalues.

Then we use bi-variate Gaussian generative modeling to do 10-class classification.

We need to optimize the likelihood function first, which means we need to optimize the parameters: π, μ_k and Σ:

$\pi = \frac{1}{n} \sum_{i=1}^{n} y_i$

$\mu_k = \frac{1}{n_k} \sum_{x_i \in C_k 1} x_i$

$\Sigma = \sum_{k=1}^{n} \frac{1}{n} \sum_{x \in C_k} (x_i - \mu_k)(x_i - \mu_k)^T$

For prediction, we plug in the parameters to:

$a_k(x) = w_k^T x + w_{k0}$

where

$w_k = \Sigma^{-1} \mu_k$

$w_{k0} = -\frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + logp(C_k)$

For each data point, we predict the value with the max $a_k(x)$.For this problem, we pick the max value in 10 results for each data point.

Result:

The training set error list for 10-fold cross validation would be: [0.23685837971552257, 0.29622758194186766, 0.32343846629560913, 0.26777983920841064, 0.272108843537415, 0.3351886209029063, 0.35435992578849723, 0.2850958565244279, 0.2683982683982684, 0.28324 056895485467]

The standard deviation of training set error list for 10-fold cross validation would be: 0.035689788528177675

The test set error list for 10-fold cross validation would be: [0.3055555555555556, 0.2722222222222222, 0.4444444444444444, 0.39444444444444443, 0.35, 0.4666666666666667, 0.39444444444444443, 0.37777777777777777, 0.31666666666666665, 0.3333333333333333]

The standard deviation of test set error list for 10-fold cross validation would be: 0.061686405471768534

Problem 4.

Q(1)

Algorithm:

Considering the two classification models, we use the discriminant model and Bayes' theorem to model $p(C|x)$:

$p(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1)+p(x|C_2)p(C_2)}$

Let a $= \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$, $p(C_1|x) = \frac{1}{1+\exp(-a)}$

Then we use maximum likelihood estimation to determine the parameters:

For one observation, the probability of obtaining the classification y is (assuming $C_1 = 1, C_2 = 0$):

$p(y|x) = p_1^y p_0^{1-y}$

Then for N independent and identical observations, the MLE is:

$\hat{w} = argmax_w\, J(w) = argmax_w \sum_{i=1}^{N} (y_i \log p_1 + (1 - y_i) \log p_0)$

Take the derivative of this function:

$\because p_1' = (\frac{1}{1+\exp(-a)})' = p_1(1 - p_1)$

$\therefore J'(w) = \sum_{i=1}^{N} y_i(1-p_1)x_i - p_1x_i + y_ip_1x_i = \sum_{i=1}^{N}(y_i - p_1)x_i$, then I use gradient descent to get the optimized w. The learning rate of the gradient descent would be 0.07 and the iteration times is 200.

Considering the multi-class classification,I use the softmax function.

$$h_\theta(x) = \begin{bmatrix} P(y = 1|x; \theta) \\ P(y = 2|x; \theta) \\ \vdots \\ P(y = K|x; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x)} \begin{bmatrix} \exp(\theta^{(1)\top} x) \\ \exp(\theta^{(2)\top} x) \\ \vdots \\ \exp(\theta^{(K)\top} x) \end{bmatrix} \tag{6}$$

where

$\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(K)} \in \Re^n$ are the parameters of the model.

Then the cost function would be:

$$J(\theta) = -\left[\sum_{i=1}^{m}(1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) + y^{(i)} \log h_\theta(x^{(i)})\right] \tag{7}$$

$$= -\left[\sum_{i=1}^{m}\sum_{k=0}^{1} 1\left\{y^{(i)} = k\right\} \log P(y^{(i)} = k|x^{(i)}; \theta)\right] \tag{8}$$

where

$P(y^{(i)} = k|x^{(i)}; \theta) = \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x^{(i)})}$

Taking derivatives, the gradient is:

$$\nabla_{\theta^{(k)}} J(\theta) = -\sum_{i=1}^{m}\left[x^{(i)}\left(1\{y^{(i)} = k\} - P(y^{(i)} = k|x^{(i)}; \theta)\right)\right] \tag{9}$$

Result:

Error list(each split for each training set percentage) of Logistic Regression for boston50 dataset: [[0.2871287128712871, 0.42574257425742573, 0.27722772277227725, 0.3564356435643564, 0.31683168316831684], [0.27722772277227725, 0.2871 287128712871, 0.3069306930693069, 0.3069306930693069, 0.45544554455445546], [0.3465346534653465, 0.27722772277227725, 0.297029702970297, 0.43564356435643564, 0.3564356435643564], [0.31683168316831684, 0.18811881188118812, 0.23762376237 62376, 0.49504950495049505, 0.27722772277227725], [0.25742574257425743, 0.25742574257425743, 0.2871287128712871, 0.4158415841584158, 0.2079207920792079], [0.1782178217821782, 0.297029702970297, 0.24752475247524752, 0.4158415841584158, 0.16831683168316833], [0.297029702970297, 0.32673267326732675, 0.3564356435643564, 0.38613861386138615, 0.48514851485148514], [0.26732673267326734, 0.26732673267326734, 0.26732673267326734, 0.5742574257425742, 0.2871287128712871], [0.2 97029702970297, 0.297029702970297, 0.3465346534653465, 0.40594059405940597, 0.4158415841584158], [0.31683168316831684, 0.18811881188118812, 0.26732673267326734, 0.37623762376237624, 0.5247524752475248]]

Error list(each split for each training set percentage) of Logistic Regression for boston75 dataset: [[0.24752475247524752, 0.24752475247524752, 0.24752475247524752, 0.5148514851485149, 0.24752475247524752], [0.22772277227722773, 0.217 82178217821782, 0.2376237623762376, 0.6138613861386139, 0.38613861386138615], [0.22772277227722773, 0.2079207920792079, 0.2376237623762376, 0.25742574257425743, 0.45544554455445546], [0.18811881188118812, 0.1782178217821782, 0.18811881 188118812, 0.18811881188118812, 0.19801980198019803], [0.16831683168316833, 0.19801980198019803, 0.21782178217821782, 0.2079207920792079, 0.18811881188118812], [0.2376237623762376, 0.2871287128712871, 0.2871287128712871, 0.217821782178 21782, 0.27722772277227725], [0.25742574257425743, 0.22772277227722773, 0.31683168316831684, 0.27722772277227725, 0.3465346534653465], [0.25742574257425743, 0.25742574257425743, 0.26732673267326734, 0.37623762376237624, 0.2673267326732 6734], [0.2376237623762376, 0.27722772277227725, 0.2376237623762376, 0.27722772277227725, 0.32673267326732675], [0.25742574257425743, 0.25742574257425743, 0.21782178217821782, 0.22772277227722773, 0.22772277227722773]]

Error list(each split for each training set percentage) of Logistic Regression for digit dataset: [[0.21448467966573817, 0.08635097493036212, 0.07520891364902507, 0.08635097493036212, 0.05013927576601671], [0.19498607242339833, 0.12534 818941504178, 0.116991643454039, 0.08913649025069638, 0.07242339832869081], [0.24233983286908078, 0.12534818941504178, 0.0807799442896936, 0.07799442896935933, 0.06406685236768803], [0.2116991643454039, 0.10863509749303621, 0.075208913 64902507, 0.07242339832869081, 0.0584958217270195], [0.23119777158774374, 0.13370473537604458, 0.10584958217270195, 0.0807799442896936, 0.06406685236768803], [0.24233983286908078, 0.12256267409470752, 0.09749303621169916, 0.07799442896 935933, 0.05292479108635097 4], [0.2479108635097493, 0.15041782729805014, 0.06685236768802229, 0.07242339832869081, 0.0584958217270195], [0.19498607242339833, 0.10027855153203342, 0.0584958217270195, 0.06128133704735376, 0.0501392757660 1671], [0.24512534818941503, 0.11420612813370473, 0.08635097493036212, 0.06963788300835655, 0.06963788300835655], [0.20891364902506965, 0.11142061281337047, 0.08913649025069638, 0.06685236768802229, 0.0584958217270195]]

mean error list(mean of the test set error rates across all splits for each training set percentage) - boston50 dataset: [0.28415842 0.28118812 0.28910891 0.41683168 0.34950495]

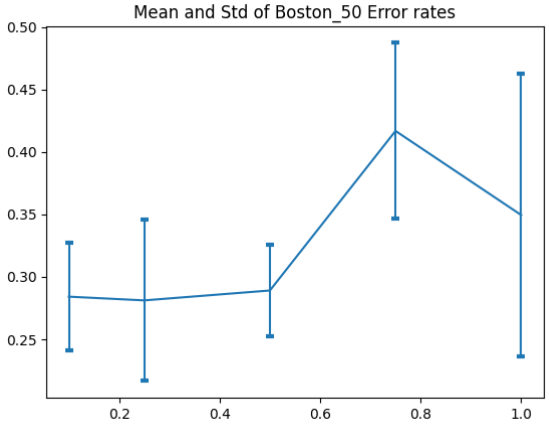
mean error list(mean of the test set error rates across all splits for each training set percentage) - boston75 dataset: [0.23069307 0.23564356 0.24554455 0.31584158 0.29207921]

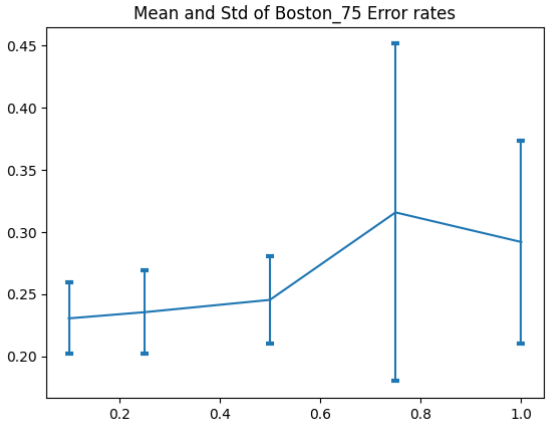
mean error list(mean of the test set error rates across all splits for each training set percentage) - digits: [0.22339833 0.1178273 0.08523677 0.07548747 0.05988858]

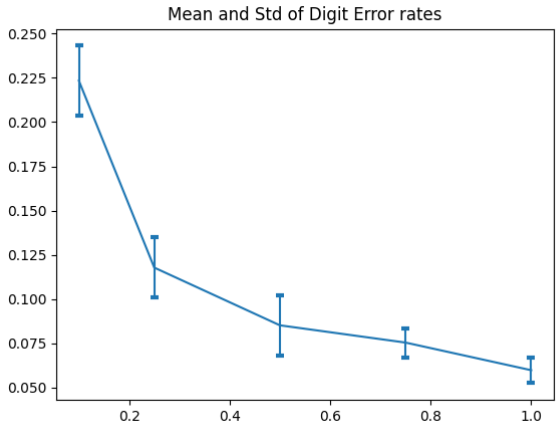
std error list(mean of the test set error rates across all splits for each training set percentage) - boston50 dataset: [0.04316877 0.06450098 0.03699315 0.07049199 0.11306653]

std error list(mean of the test set error rates across all splits for each training set percentage) - boston70 dataset: [0.02871287 0.03366337 0.0350892 0.13557128 0.08143527]

std error list(mean of the test set error rates across all splits for each training set percentage) - digits: [0.01972806 0.01699164 0.01695278 0.00821137 0.00729039]







Q(2)

Algorithm:

For Naive-Bayes, One of the simplest assumptions is the conditional independence:

$$p(x|y) = \prod_{i=1}^{p} p(x_i|y)$$

So using Bayes' theorem, for a single observation:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} = \frac{\prod_{i=1}^{p} p(x_i|y)p(y)}{p(x)}$$

Also, we need to make further assumptions about the conditional probability of a single dimension and the class prior:

1. x_i is a continuous variable: $p(x_i|y) = \mathcal{N}(\mu_i, \sigma_i^2)$

2. x_i is a discrete variable: Categorical:$p(x_i = i|y) = \theta_i, \sum_{i=1}^{K} \theta_i = 1$

$\therefore$ for K-classes, we have $a_k(x) = \sum_{i=1}^{p}(x_i log\mu_{ik} + (1 - x_i)log(1 - \mu_{ik})) + logp(C_k)$

where

$log\mu_{ik} = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu)^2 - \frac{N}{2}log\sigma^2 - \frac{N}{2}log(2\pi)$, $p(C_k)$ is the prior probability, μ and σ^2 are the matrixes containing the mean and variance for each

feature(13,64) of each output(0-1,0-9)

We do not need to calculate the constant term because it won't affect our results.For each data point, we predict the value with the max $a_k(x)$.For this problem, we pick the max value in 2 or 10 results for each data point.

Result:
Error list(each split for each training set percentage) of Naive Bayes for boston50 dataset: [[0.36633663366336633, 0.33663366336633666, 0.3069306930693069, 0.26732673267326734, 0.26732673267326734], [0.26732673267326734, 0.2079207920 792079, 0.1485148514851485, 0.13861386138613863, 0.27722772277227725], [0.27722772277227725, 0.21782178217821782, 0.26732673267326734, 0.3069306930693069, 0.24752475247524752], [0.26732673267326734, 0.1485148514851485, 0.23762376237623 76, 0.25742574257425743, 0.16831683168316833], [0.42574257425742573, 0.18811881188118812, 0.24752475247524752, 0.15841584158415842, 0.21782178217821782], [0.24752475247524752, 0.2079207920792079, 0.19801980198019803, 0.1881188118811881 2, 0.2079207920792079], [0.3069306930693069, 0.2871287128712871, 0.21782178217821782, 0.21782178217821782, 0.24752475247524752], [0.39603960396039606, 0.2871287128712871, 0.24752475247524752, 0.2079207920792079, 0.22772277227722773], [0.3564356435643564, 0.24752475247524752, 0.21782178217821782, 0.18811881188118812, 0.25742574257425743], [0.24752475247524752, 0.21782178217821782, 0.19801980198019803, 0.13861386138613863, 0.22772277227722773]]

Error list(each split for each training set percentage) of Naive Bayes for boston75 dataset: [[0.26732673267326734, 0.21782178217821782, 0.2079207920792079, 0.16831683168316833, 0.25742574257425743], [0.2079207920792079, 0.22772277227 722773, 0.297029702970297, 0.18811881188118812, 0.2871287128712871], [0.32673267326732675, 0.1782178217821782, 0.2079207920792079, 0.15841584158415842, 0.25742574257425743], [0.22772277227722773, 0.1782178217821782, 0.26732673267326734 , 0.15841584158415842, 0.297029702970297], [0.18811881188118812, 0.13861386138613863, 0.1485148514851485, 0.10891089108910891, 0.21782178217821782], [0.27722772277227725, 0.25742574257425743, 0.24752475247524752, 0.19801980198019803, 0 .22772277227722773], [0.25742574257425743, 0.21782178217821782, 0.21782178217821782, 0.18811881188118812, 0.24752475247524752], [0.1782178217821782, 0.19801980198019803, 0.18811881188118812, 0.15841584158415842, 0.27722772277227725], [0.297029702970297, 0.2079207920792079, 0.2079207920792079, 0.19801980198019803, 0.25742574257425743], [0.2376237623762376, 0.24752475247524752, 0.3465346534653465, 0.24752475247524752, 0.31683168316831684]]

Error list(each split for each training set percentage) of Naive Bayes for digits dataset: [[0.28690807799442897, 0.2116991643454039, 0.13649025069637882, 0.10027855153203342, 0.08356545961002786], [0.30919220055710306, 0.189415041782 7298, 0.12813370473537605, 0.10027855153203342, 0.08635097493036212], [0.37604456824512533, 0.2200557103064067, 0.17548746518105848, 0.1392757660167131, 0.116991643454039], [0.3008356545961003, 0.18662952646239556, 0.11420612813370473, 0.10306406685236769, 0.08356545961002786], [0.26740947075208915, 0.16991643454038996, 0.116991643454039, 0.08913649025069638, 0.10863509749303621], [0.34818941504178275, 0.19220055710306408, 0.14763231197771587, 0.09192200557103064, 0 .08913649025069638], [0.31754874651810583,

0.181058495821727, 0.13370473537604458, 0.11142061281337047, 0.11142061281337047], [0.362116991643454, 0.20055710306406685, 0.1309192200557103, 0.11142061281337047, 0.09749303621169916], [0.31 47632311977716, 0.16991643454038996, 0.09749303621169916, 0.09749303621169916, 0.10027855153203342], [0.3231197771587744, 0.2116991643454039, 0.15877437325905291, 0.12813370473537605, 0.13649025069637882]]

Mean error list(mean of the test set error rates across all splits for each training set percentage) - boston50 dataset: [0.31584158 0.23465347 0.22871287 0.20693069 0.23465347]

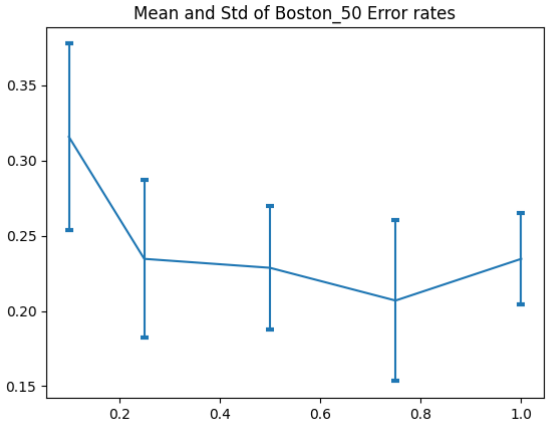
Mean error list(mean of the test set error rates across all splits for each training set percentage) - boston70 dataset: [0.24653465 0.20693069 0.23366337 0.17722772 0.26435644]

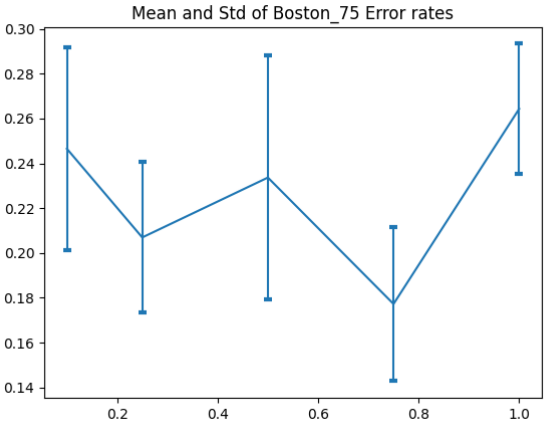
Mean error list(mean of the test set error rates across all splits for each training set percentage) - digits: [0.32061281 0.19331476 0.13398329 0.10724234 0.10139276]

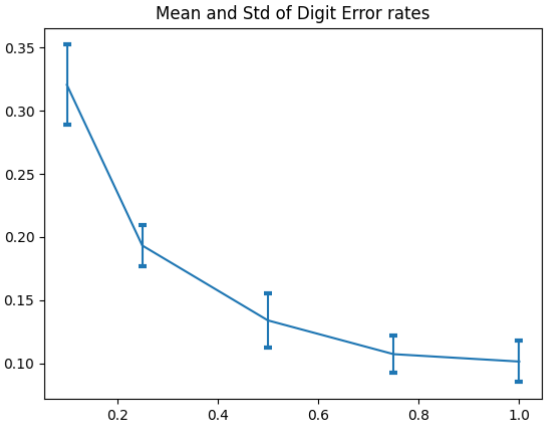
Std error list(mean of the test set error rates across all splits for each training set percentage) - boston50 dataset: [0.06190297 0.05258721 0.04116955 0.05358439 0.030372]

Std error list(mean of the test set error rates across all splits for each training set percentage) - boston70 dataset: [0.04525305 0.03356129 0.05444644 0.03442641 0.02905228]

Std error list(mean of the test set error rates across all splits for each training set percentage) - digits: [0.03177075 0.01658259 0.02141226 0.01506498 0.01629943]




Mean and Std of Boston_75 Error rates
0.30
0.28
0.26
0.24
0.22
0.20
0.18
0.16
0.14
0.2
0.4
0.6
0.8
1.0


Mean and Std of Digit Error rates
0.35
0.30
0.25
0.20
0.15
0.10
0.2
0.4
0.6
0.8
1.0

Reference:

1. http://deeplearning.stanford.edu/tutorial/supervised/SoftmaxRegression/

2.Pattern Recognition and Machine Learning,CHRISTOPHER M.BISHOP

3.Lecture in Canvas,Nicholas Johnson