CSCI5525 HW1

jude0010

February 2022

1 Problem 1

(a) We want to find an $f(x)$ such that $f(x)$ is a stationary point of our loss function $L(f(x))$. I will use the result from the calculus of variations that given a functional defined by an integral over a function $G(f, f', x)$, i.e.

$$L(f) = \int G(f(x), f'(x), x)dx$$

, the stationary points of the functional are given by the Euler-Lagrange equations, i.e.

$$\frac{\partial G}{\partial f} - \frac{d}{dx}(\frac{\partial G}{\partial f'}) = 0$$

. In our example,

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

. Then,

$$G = \int l(f(x), y)p(x, y)dy$$

$$= p(x) \int (f(x) - y)^2 p(y|x)dy$$

So, by application of the Euler-Lagrange equation, linearity of the derivative operator, and because the above integral is not with respect to f,

$$p(x) \int 2(f(x) - y)p(y|x)dy = 0$$

$$\int (f(x) - y)p(y|x) = 0$$

$$\int f(x)p(y|x)dy = \int yp(y|x)dy$$

Let's analyze the left and right hand sides of the above equation.

Left hand side: $f(x)$ does not depend on y and the total probability that y attains some value out of its possible values is one no matter what this event is conditioned on (in this case x). So, the left hand side is $f(x)$.

Right hand side: The integral of the values of a random variable weighted by its probabilities is the expectation of the random variable. The integral of the values of a random variable weighted by its probabilities conditioned on x is the expectation of the random variable conditioned on x. So, the right hand side is $E_y[y|x]$.

So,

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

is the optimal $f(x)$ for minimizing for least squares loss, which is the conditional expectation of y given x.

(b) We want to find an $f(x)$ such that $f(x)$ is a stationary point of our loss function $L(f(x))$. I will use the result from the calculus of variations that given a functional defined by an integral over a function $G(f, f', x)$, i.e.

$$L(f) = \int G(f(x), f'(x), x)dx$$

, the stationary points of the functional are given by the Euler-Lagrange equations, i.e.

$$\frac{\partial G}{\partial f} - \frac{d}{dx}(\frac{\partial G}{\partial f'}) = 0$$

. In our example,

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

. Then,

$$G = \int l(f(x), y)p(x, y)dy$$

$$= p(x) \int |f(x) - y|p(y|x)dy$$

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

So, by application of the Euler-Lagrange equation and the Leibniz integral rule,

$$\frac{d}{df(x)}p(x)(\int_{-\infty}^{f(x)} (f(x) - y)p(y|x)dy - \int_{f(x)}^{\infty} (f(x) - y)p(y|x)dy) = 0$$

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

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

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

Consider the above equality:

The probability that the random variable y attains some range of values, $p(a \leq y \leq b)$, is equal to the corresponding area of the pdf of the random variable y, i.e. $\int_a^b f_y(u)du$. So, given some x, the above equality is true for $f(x)$ such that:

$$\int_{-\infty}^{f(x)} f_{y|x}(u)du = \int_{f(x)}^{\infty} f_{y|x}(u)du$$

. By definition then, $f(x)$ is the median of y given x. Q.E.D.

2 Problem 2

Yes, Professor Bayes is correct.

Think of the current classifier as labelling regions of the domain of x with predicted target labels.

So, there are two regions R^- and R^+ that partition the domain of x such that $x \in R^+$ if and only if $P(y = +1|x) \geq \frac{1}{2}$. Then, since R^+ and R^- partition the domain of x, $x \in R^-$ if and only if $P(y = +1|x) < \frac{1}{2}$. So, since there are only two available labels for y, namely $+1$ and -1, $x \in R^-$ if and only if $P(y \neq +1|x) = P(y = -1|x) > \frac{1}{2}$. Then, for any x in R^+, $P(y \neq +1|x) = P(y = -1|x) \leq \frac{1}{2}$. Additionally, for any x in R^+, $f^*(x) = +1$, while for any x in R^-, $f^*(x) = -1$.

I will first use proof by cases to show that Professor Bayes is correct when we restrict our attention to classifiers $f'(x)$ that differ from $f^*(x)$ for exactly one x, which I call x'. Consider two cases:

Case 1: $f'(x)$ is a classifier that differs from $f^*(x)$ exactly in that it classifies x' in R^+ as -1. Then,

$$L(f'(x)) = L(f^*(x)) + P(x')(-P(y \neq +1|x') + P(y \neq -1|x'))$$

$$P(x') \geq 0$$

$$P(y \neq -1|x') \geq \frac{1}{2} \geq P(y \neq +1|x')$$

$$L(f'(x)) \geq L(f^*(x))$$

.

Case 2: $f'(x)$ is a classifier that differs from $f^*(x)$ exactly in that it classifies x' in R^- as $+1$. Then,

$$L(f'(x)) = L(f^*(x)) + P(x')(-P(y \neq -1|x') + P(y \neq +1|x'))$$

$$P(x') \geq 0$$

$$P(y \neq +1|x') > \frac{1}{2} > P(y \neq -1|x')$$

$$L(f'(x)) \geq L(f^*(x))$$

.

So, by proof by cases, : $(f'(x)) \geq L(f^*(x))$.

Additionally, notice that the change in expected loss when the predicted label associated with x' changed did not depend on the classifier's predictions for other values of x. Therefore, this result holds for a single reclassification of x' in R^+ from $+1$ to -1 and for a single reclassification of x' in R^- from -1 to $+1$, regardless of the classifier's other properties. Since any new classifier f can be reached by a series of reclassifications of individual points of the above two types, each of which can not reduce loss, the loss of any new classifier f can not be less than $L(f^*(x))$. Q.E.D.

3 Problem 3

(i) average error on training set across all folds in LDA1dThres: 0.14360323886639675

average error on test set across all folds in LDA1dThres: 0.16207843137254901

standard deviation of error on training set across all folds in LDA1dThres: 0.011543755959575764

standard deviation of error on test set across all folds in LDA1dThres: 0.06287120904941816

Summary of methods:

1-dimensional case:

I wanted to choose a weight vector such that $w^T x$ becomes the representation of x in the new feature space. To do this, I wanted to choose a weight vector w to maximize the ratio of between-class variance to within-class variance. I represented between class separation as $(m_2 - m_1)^2$, the square of the difference of the means of the two classes. I represented within class separation as $s_1^2 + s_2^2$, the sum of the variances within the two classes. I maximized this by choosing a weight vector to maximize the inverse of the within-class covariance matrix dotted with the difference of the means of the two classes. The within-class covariance matrix, S_W, is given by

$$S_W = \sum_{n \in C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_2} (x_n - m_2)(x_n - m_2)^T$$

. To create the threshold, I chopped between the means of the two classes, weighting the chop so that the threshold was closer to the mean for whichever class had less samples. In other words, the portion of the region between the means associated with class i was proportional to the prior probability of class i. To predict, I dotted a new x with the weight vector and compared this value to the threshold to make an assignment.

(5 point problem)

Yes, you can because the number of dimensions you are projecting to is less than the number of dimensions in the original feature space. $S_W^{-1} S_B$ will have a number of independent eigenvectors equal to the number of dimensions in the original feature space, so it is just a matter of selecting the two eigenvectors with the largest eigenvalues to create W. Additionally, all of the attributes here seem like they could be reasonably assumed to be approximately normal without significant outliers. Additionally, there are sufficient numbers of elements of each class relative to the number of attributes. Overall, there are no issues with this data set that would render LDA unusable for two dimensions.

(ii) average error on training set across all folds in LDA2dGaussGM: 0.43714706918839

average error on test set across all folds in LDA2dGaussGM: 0.4479577901924271

standard deviation of error on training set across all folds in LDA2dGaussGM: 0.023094043707599883

standard deviation of error on test set across all folds in LDA2dGaussGM: 0.041512631135625234

Summary of methods:

2-dimensional case:

Here, the weight matrix W had two columns. The projections were two-dimensional, $y = W^T x$. We this time found the two eigenvectors of $S_W^{-1} S_B$ associated with the largest eigenvalues of that matrix. S_W here was generalized to be

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

, where

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

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

. These vectors became the columns of W. Then, a Gaussian bivariate normal distribution was associated with the resulting projections of the ten classes. To estimate the parameters, I used the maximum likelihood estimations

$$\hat{\mu}_n = \frac{1}{n}\sum_{j=1}^{n} x_j$$

$$\hat{V}_n = \frac{1}{n}\sum_{j=1}^{n}(x_j - \hat{\mu}_n)(x_j - \hat{\mu}_n)^T$$

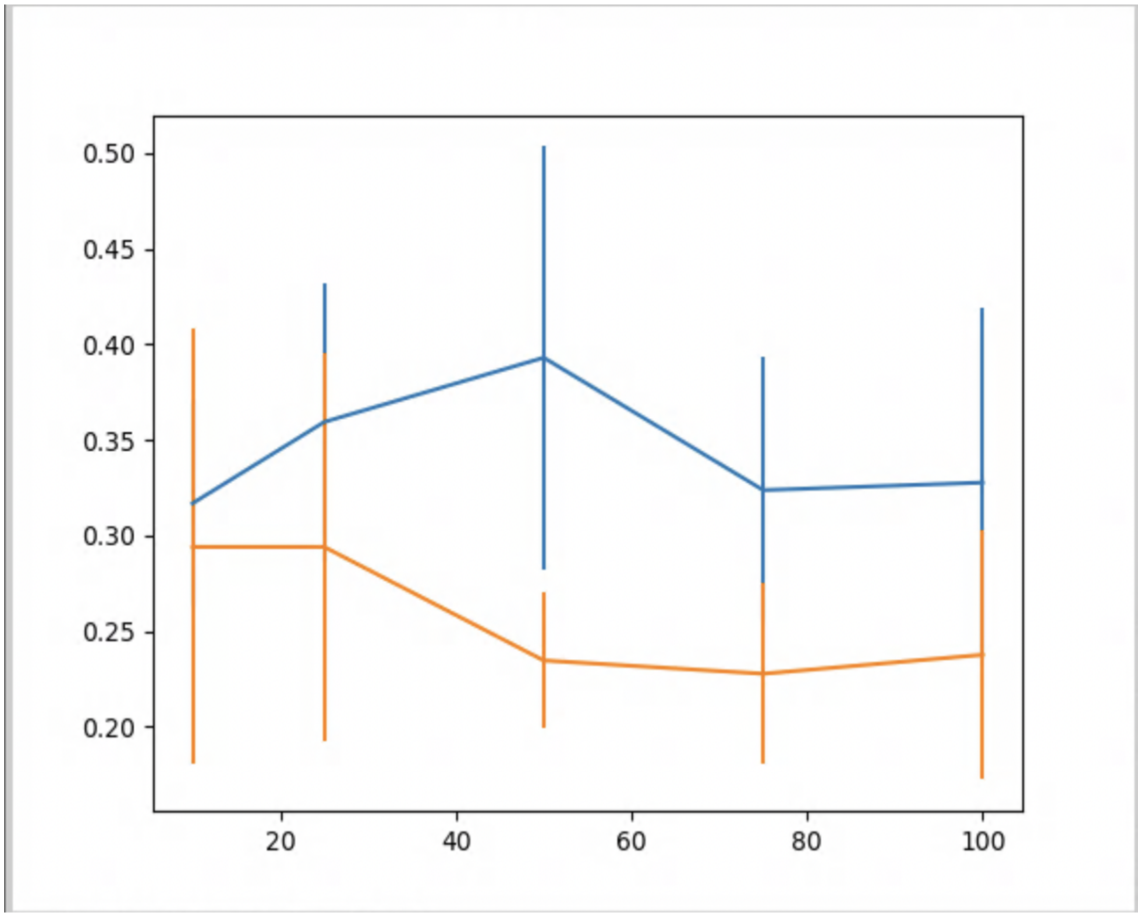
. Then, for prediction on a new sample, a relative likelihood was associated with each class for that sample, where this likelihood was proportional to the natural logarithm of the product of the prior probability of the class and the likelihood of the sample given the estimated Gaussian probability distribution of that class. The probability of x is independent of class, so that term was dropped, and natural logarithm is a monotonically increasing function, so it preserves orderings. The relative likelihood for each class was given by

$$l_k = ln(p(C_k)) + ln(\frac{1}{2\pi}) - \frac{1}{2}ln(|\Sigma|) - \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)$$

. The class with the highest l was predicted to be the actual class for each test point in order to minimize the error rate of the classifier on the test set.

4 Problem 4

(i) The below graph shows the means and standard deviations of the error rates for Logistic Regression on Boston50 and Boston75 for increasing percentages of the training data set.



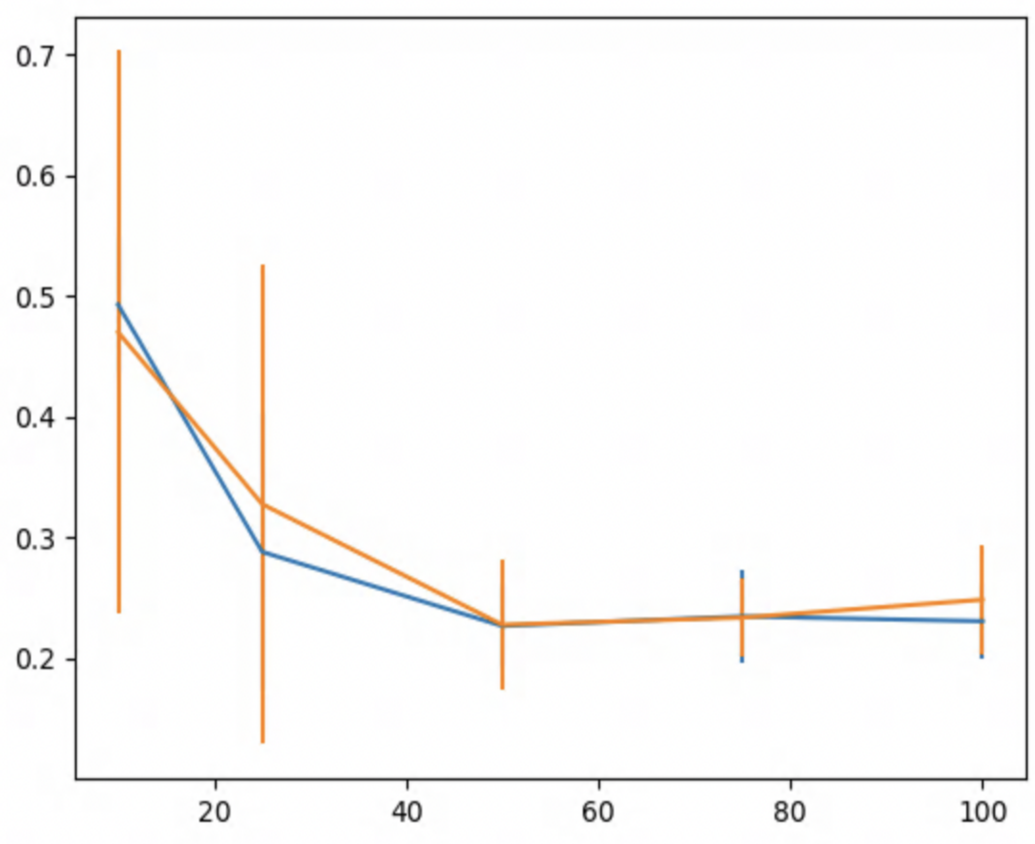
Summary of methods:

My version of Logistic Regression only works for binary classification. It uses mini-batch gradient descent, which means that in my case the data is split into groups of 100 and updates occur due to each of these batches within each epoch. 500 epochs are used for learning. The sigmoid activation function is used on $Xw + b$, where b, the bias, I think of as a threshold. Weights are updated in each epoch to reduce loss, which is given implicitly by

$$\sum_{n=1}^{N}(1 - y_n)ln(1 - \hat{y}_n) - y_n ln(\hat{y}_n)$$

. This is done using gradient descent.

(ii) The below graph shows the means and standard deviations of the error rates for Naive Bayes' on Boston50 and Boston75 for increasing percentages of the training data set.



Summary of methods:

Here, the weight matrix W had d columns. A Gaussian d-dimensional normal distribution was associated with each of the ten classes. To estimate the parameters, I used the maximum likelihood estimations

$$\hat{\mu}_n = \frac{1}{n}\sum_{j=1}^{n} x_j$$

$$\hat{V}_n = \frac{1}{n}\sum_{j=1}^{n}(x_j - \hat{\mu}_n)(x_j - \hat{\mu}_n)^T$$

. Then, for prediction on a new sample, a relative likelihood was associated with each class for that sample, where this likelihood was proportional to the natural logarithm of the product of the prior probability of the class and the likelihood of the sample given the estimated Gaussian probability distribution of that class. The probability of x is independent of class, so that term was dropped, and natural logarithm is a monotonically increasing function, so it preserves orderings. The relative likelihood for each class was given by

$$l_k = ln(p(C_k)) + \frac{d}{2}ln(\frac{1}{2\pi}) - \frac{1}{2}ln(|\Sigma|) - \frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)$$

. The class with the highest l was predicted to be the actual class for each test point in order to minimize the error rate of the classifier on the test set.