CSE 575: Statistical Machine Learning Assignment #2
Sushant Trivedi (1213366971)

1. LOGISTIC REGRESSION

How many independent parameters for Model 1? What are they?
There are 3 independent parameters for Model 1, namely, w1, w2, w3

How many independent parameters for Model 2? What are they?
There are 4 independent parameters for Model 2, namely w0, w1, w2, w3

Does it matter how the third example is labeled in Model 1? Does it matter in Model 2? Please justify your answers.

In Model 1, as all the feature values are equal to 0 their contribution to weight by gradient descent will be irrelevant for example 3. Thus, it doesn't affect the weights by training

In Model 2, it will affect the weights during training procedure because it has a bias term w0 whose value will be affected even when all the features are equal to zero.

For Model 1, we want to maximize the conditional log-likelihood with an L2 regularization term (we can use λ as the penalized parameter) What would the conditional log-likelihood term behave when λ is very large? Can you derive the MLE of the parameter w in terms of λ and the training data $X^{(1)}, Y^{(1)}$. Based on this, can you explain how the weights w will behave as λ increases.

$$l(w) = \sum_i \log\left(P(y_i|x_i, w) - \frac{\lambda}{2}||w||^2\right) = \sum_i \log(f(y_i w^T x_i) - \frac{\lambda}{2}||w||^2$$

Please note that the first term can be approximated for a linear function of w for Large λ values Thus,

$$\textstyle\sum_i \log(f(y_i w^T x_i) \approx \sum_i \frac{1}{2} y_i w^T x_i \quad \text{for each sample i ranging from 1 to N}$$

Therefore,

$$l(w) \approx \sum_i \frac{1}{2} y_i w^T x_i - \frac{\lambda}{2}||w||^2$$

Taking differentiation w.r.t to w1, w2, w3, we get

$$\frac{d(l(w))}{dw_j} \approx \frac{1}{2}\sum_i y_i\, x_i^j - \lambda w_j$$

Where,

$i \rightarrow sample\ no, j \rightarrow feature\ no$
$x_i^j \rightarrow is\ the\ j^{th}\ feature\ value\ for\ the\ i^{th}\ sample$
$w_j \rightarrow is\ the\ weight\ for\ feature\ x^j$

Solving the above equation we would get

$$w = \frac{1}{2\lambda} \sum_i y_i x_i$$

Conclusion
Parameter λ controls the impact of the regularization term. Higher λ result in smaller weights, but too high values for λ lead to underfitting.

2. <u>CONVEXITY OF LOGISTIC REGRESSION</u>

<u>Proof 1:</u>

$$l(w) = \underbrace{\sum_j y^j (w_0 + \sum_{i=1}^{d} w_i x_i^j)}_{\textbf{Function 1}} - \underbrace{ln(1 + exp(w_0 + \sum_{i=1}^{d} w_i x_i^j))}_{\textbf{Function 2}}$$

Sum of concave function are concave. By definition, Function 1 is a linear function which is also a concave function. Function 2 is log(exp()) of a linear function which is also concave.
Thus, log conditional likelihood function l(w) is concave as it's a sum of 2 concave function.

<u>Proof 2:</u>
We can see that the double derivative of the log conditional likelihood function l(w) would be negative. Thus, it implies that l(w) is concave. The same can be done by finding the Hessian matrix of the error function, which happens to be semi-definite. Thus, the l(w) function is concave by definition
[Refer Pg 207 from Pattern Recognition and Machine Learning, Bishop]

$p(x) \rightarrow sigmoid\ of\ x$

$$l(w) = \log\left(\prod_i^N p(x_i)^{y_i} \big(1 - p(x_i)\big)^{1-y_i} \right)$$

As simplified in class,

$$l(w) = \sum_i^N (y_i - 1) w x_i + \log(1 + e^{-w x_i})$$

Taking single derivative we get

$$\frac{dl(w)}{dw} = \sum_i^N (y_i - 1) x_i + \left(\frac{e^{-w x_i}}{1 + e^{-w x_i}} \right) x_i$$

Take another derivative to get

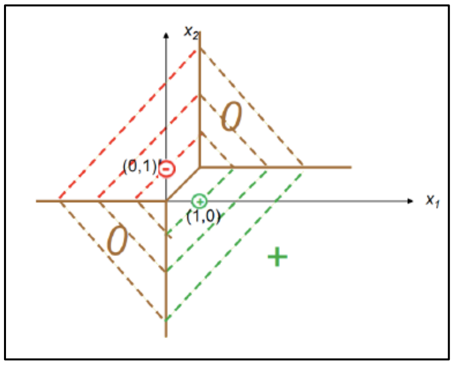
$$\frac{d^2 l(w)}{dw^2} = \sum_i^N -\frac{x_i dp(x_i)}{dw} = \sum_i^N \frac{-x_i^2 e^{-wx_i}}{(1+e^{-wx_i})^2}$$

Conclusion

Observing from the equation we see that x_i^2 ***and*** $\frac{e^{-wx_i}}{(1+e^{-wx_i})^2}$ **are always positive**.

Therefore, $\frac{d^2 l(w)}{dw^2}$ **is always negative**. Hence, the log conditional likelihood function in logistic regression is always concave

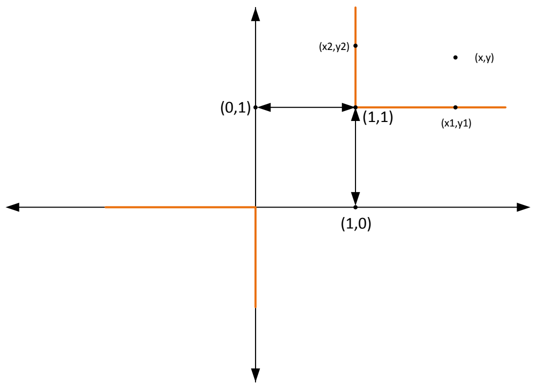
3. <u>1NN CLASSIFIER DECISION BOUNDARY</u>



a.) Prove that the decision boundary of 1NN classifier with L₁ distance is the one as shown in Figure 1 (i.e., we classify all the examples from the green area as positive; all the examples from the red area as negative; and the brown area is the decision boundary).

With the L1 norm (taxicab distance), the decision boundary lies on points for which the L1 norm for (1,0) = L1 norm for (0,1).

L1 norm for (x,y) and (1,0) = |x-1|+|y|
L1 norm for (x,y) and (0,1) = |x|+|y-1|
Equating the two we are able to prove that the regions marked in brown is the decision boundary.



<u>**For (x1, y1)**</u>
L1 norm for (1,0) = x1
L1 norm for (0,1) = x1

<u>**For (x2, y2)**</u>
L1 norm for (1,0) = y2
L1 norm for (0,1) = y2

<u>**For (x,y)**</u>
L1 norm for (1,0) = 1+(y-1)+(x-1) = x+y-1
L1 norm for (0,1) = 1+(x-1)+(x-1) = x+y-1

Similarly, for the lower half we can show that the L1 norm = x+y+1
Similarly, for red and green area we can show that they are lower L1 norm with (0,1) and (1,0) respectively.

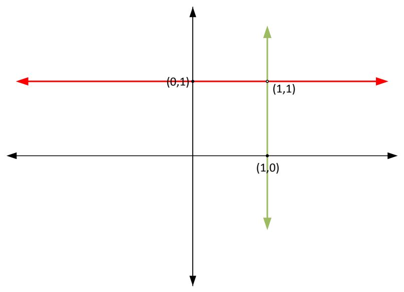
Hence the brown part can be shown as the decision boundary, red part as negative and green part as positive

b.) What will be the decision boundary if we use L_2 distance instead?

L2 norm is the Euclidean distance between two points. ***Thus, the L2 decision boundary would be line x=y.***

c.) What will be the decision boundary if we use L_0 distance instead?

L0 norm is count of non-zero elements in a vector. In this case (x,y) co-ordinate vector.



As per this, any point on red line is the classified as negative and on green line as positive as the L0 norm is 1 w.r.t (0,1) and (1,0) respectively except point (1,1).

On any other point on the plane and (1,1) point, the L0 norm is equal to 2 for both (1,0) and (0,1). Hence, they are the decision boundary.

d.) What will be the decision boundary if we use L_∞ distance instead?

L_∞ norm is max (Δx, Δy) between 2 points.
Thus,
L_∞ norm for (1,0) = max (|x-1|, |y|)
L_∞ norm for (0,1) = max (|x|, |y-1|)

Equating these two we find that the decision boundary is x=y.

4. <u>THE MARGIN OF SVM</u>

Prove that the size of the margin of the given classifier is equal to $\frac{2}{||w||}$

Linear Classifier: $w'x + b = 0$
$w'x + b > 0$ for +1 class
$w'x + b < 0$ for -1 class
$d^{+} \rightarrow This\ is\ the\ nearest\ element\ in + 1\ class.$
$d^{-} \rightarrow This\ is\ the\ nearest\ element\ in - 1\ class.$

Total margin size for the classifier = $m = d^{+} + d^{-}$

$$m \ = \frac{< w, X^{+} >}{||w||} + \frac{< w, X^{-} >}{||w||}$$
$$m = \frac{1+d}{||w||} - \frac{-1-d}{||w||}$$
$$m \ = \frac{1}{\left||w|\right|} + \frac{d}{\left||w|\right|} + \frac{1}{\left||w|\right|} - \frac{d}{\left||w|\right|}$$

$$\boldsymbol{margin\ (m)} = \frac{\mathbf{2}}{||\boldsymbol{w}||}$$

5. <u>KERNELIZED SVM</u>

a.) What is the corresponding kernel $\boldsymbol{K(X^{i}, X^{j})}$?

As it is a polynomial, Kernel $K\left(X^{i}, X^{j}\right) = \left(1 + \left(X^{i} X^{j}\right)\right)^{2}$ as mentioned in the presentations "Classification" slide no 80.

b.) Suppose we want to train a hard-margin linear SVM in this mapped feature space. What is the Langrangian dual problem?

<u>Langrangian dual problem</u>

$max\ Q(\alpha) = \ \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}(9\alpha_1^2 - 2\alpha_1\alpha_2 - \ 2\alpha_1\alpha_2 - \ 2\alpha_1\alpha_4 - \ 9\alpha_1^2 + 2\alpha_2\alpha_3 -$
$2\alpha_2\alpha_4 + 9\alpha_3^2 + 2\alpha_3\alpha_4 + 9\alpha_4^2)$

Such that $\alpha_i \geq 0\ for\ all\ i = 1,2,3,4$

$$\alpha_1 + \alpha_4 = \alpha_2 + \alpha_3$$

c.) Suppose we use SMO algorithm to solve the above optimization problem. We fix α1 = α4 = 1=4 and update α2 and α3. What are the updated α2 and α3, respectively?

d.) Suppose we use the above αi (i = 1; 2; 3; 4) as the final solution for the Lagrangian dual problem. What is the weighted vector w? What is the off-set scalar b? What is the decision boundary of your SVM classifier?

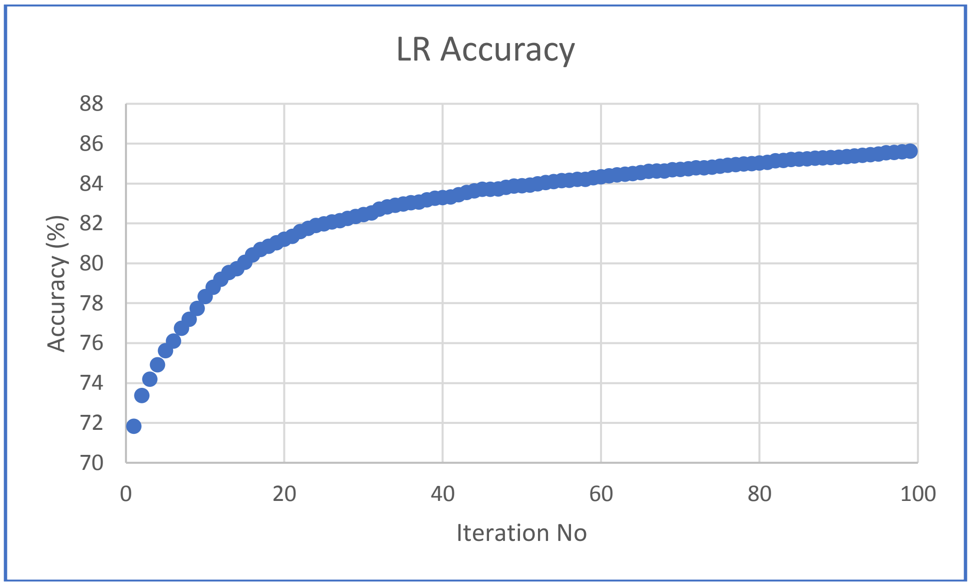
$$w = \sum_{i}^{4} \alpha_i y_i \emptyset(x_i) = \left[0,0,-\frac{1}{\sqrt{2}},0,0,0\right]'$$

b = 0

The decision boundary is $X_1 X_2 = 0$, this means inverted classification. That is positive is negative label and negative is positive label.

6. <u>MNIST HANDWRITTEN DIGITS CODE</u>

a) LOGISTIC REGRESSION

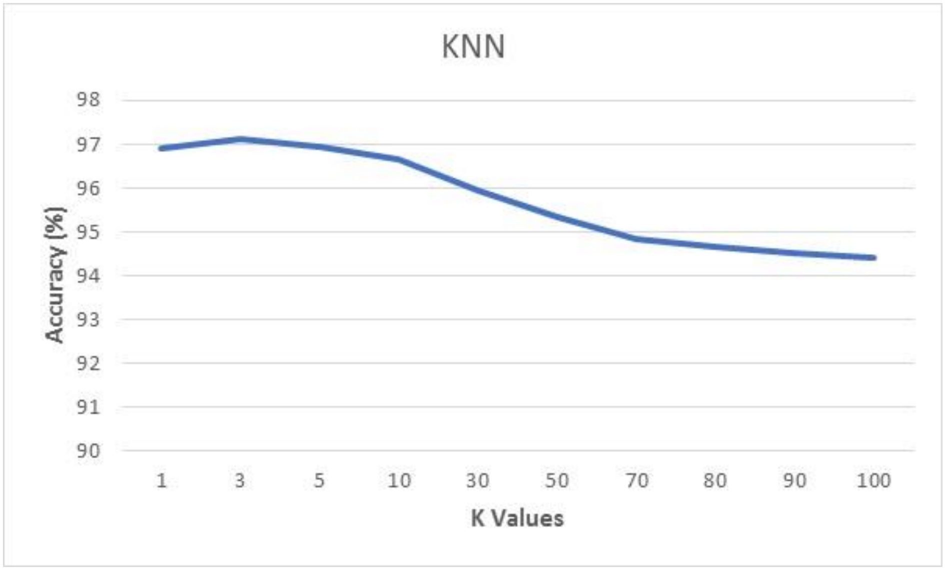


By playing around with learning rate factor I was able to change the percentage accuracy achieved within 100 iterations. It was observed that depending on this the accuracy steadily increases or may show oscillatory behavior as it may go down it little bit.

For my implementation, the accuracy saturates at 85% accuracy as can be seen in the graph above.

b) KNN METHOD

k	Accuracy
1	96.91
3	97.14
5	96.94
10	96.66
30	95.94
50	95.35
70	94.85
80	94.65
90	94.51
100	94.4



Here we observe that the for k=1 we have large initial accuracy as we are overfitting our model to our training data. And this accuracy decreases as the k values increases.