Sarthak Maharana

EE 559

Homework 4.

1. The criterion function of a 2-class perception algorithm is

$$J(\underline{w}) = -\sum_{n=1}^{N} [[\underline{w}^T z_n \underline{x_n} \leq 0]] \, \underline{w}^T z_n \underline{x_n}$$

This can be written as follows:

$$J(\underline{w}) = \begin{cases} -\sum_{n=1}^{N} \underline{w}^T z_n \underline{x_n} & \text{if } \underline{w}^T z_n \underline{x_n} \leq 0 \\ 0 & \text{if } \underline{w}^T z_n \underline{x_n} > 0 . \end{cases}$$

$$= \max\left(-\sum_{n=1}^{N} \underline{w}^T z_n x_n , 0\right)$$

Now, the standard definition of a convex function is as follows: A function $f: \mathbb{R}^n \to \mathbb{R}$ is convex, if the domain of f is convex and $\forall x, y \in$ domain f and $0 \leq \theta \leq 1$ st $f(\theta x + (1-\theta) y) \leq \theta(f(x)) + (1-\theta) f(y)$.

$$J(\underline{w}) = \max\left(-\sum_{n=1}^{N} \underline{w}^T z_n x_n , 0\right).$$

Let $f(\underline{w}) = -\sum_{n=1}^{N} \underline{w}^T z_n x_n$ and $g(\underline{w}) = 0 \;\forall\; w_1, w_2 \in \mathbb{R}^n$

and $\theta \in (0,1)$.

Now, $f(\theta w_1 + (1-\theta) w_2) \leq \theta f(w_1) + (1-\theta) f(w_2)$. — ①

$f(w) \leq J(w) = \max(f(w), g(w))$

① can be written as:

$f(\theta(w_1) + (1-\theta) w_2) \leq \theta J(w_1) + (1-\theta) J(w_2)$. — ②

Similarly, $g(\theta w_1 + (1-\theta) w_2) \leq \theta g(w_1) + (1-\theta) g(w_2)$

$\leq \theta J(w_1) + J(w_2)(1-\theta)$. — ③

We made an assumption that $f(w) = -\sum_{n=1}^{N} \underline{w}^T z_n \underline{x}_n$ and $g(w)$ are convex.

and so, from ② and ③,

$$J(w_1 + (1-\theta) w_2) \leq \theta J(w_1) + (1-\theta) J(w_2).$$

or, from ② and ③

$$J(\theta w_1 + (1-\theta) w_2) = \max\{f(\theta w_1 + (1-\theta) w_2), g(\theta w_1 + (1-\theta) w_2)\}$$
$$\leq \theta J(w_1) + (1-\theta) J(w_2) \quad // \qquad \theta \in (0,1).$$

which shows that the max function $J(\underline{w}) = \max(f(w), g(w))$ is convex, where

$$f(\underline{w}) = -\sum_{n=1}^{N} w^T z_n x_n \quad \text{and} \quad g(\underline{w}) = 0 \quad //$$

HOMEWORK 4

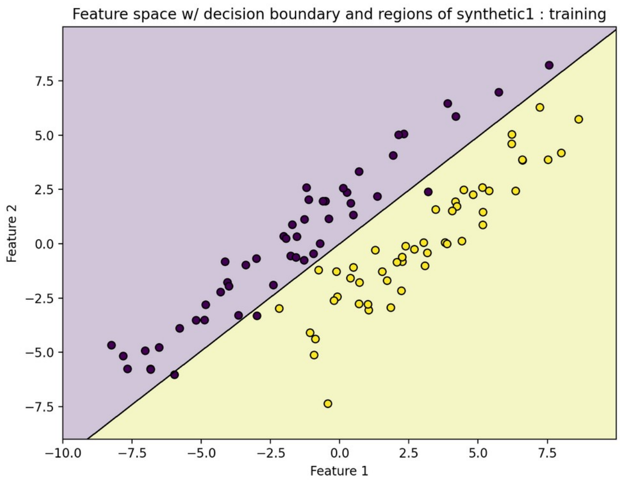
2. We train the perceptron classifying algorithm on the training set of each dataset to obtain optimal weights and the final value of the criterion function. Then, the optimal weights are used to make predictions on the test data of the corresponding dataset.

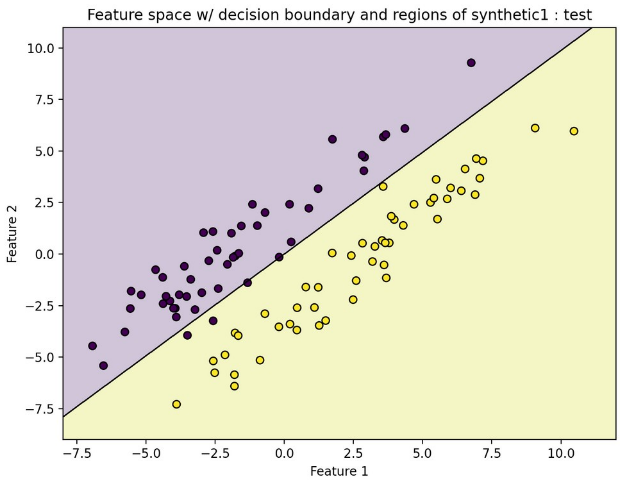
(a) The following are the results on the “synthetic1” dataset:

(i) The optimal weights $\underline{\hat{w}}$, i.e the global minima = **[-4.820879, 4.881458]**. The algorithm halted without convergence, with a final **J($\underline{\hat{w}}$) = 4.19378.**

(ii) After running the perceptron classifier on the training set and the test set, we obtain classification errors of **0.03 and 0.03**, respectively.

(iii) The following two figures illustrate the feature space of the training and test data set of synthetic1. Since we’re dealing with two features, “feature 1” and”feature 2” are plotted on the x and y axes respectively. As expected, we can see roughly 3 misclassifications on both the sets.



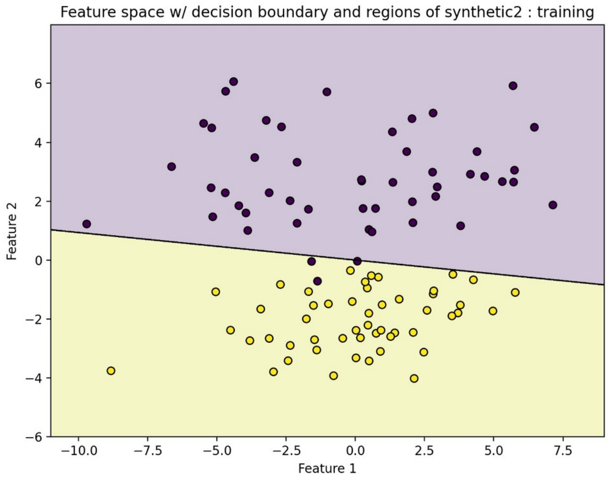


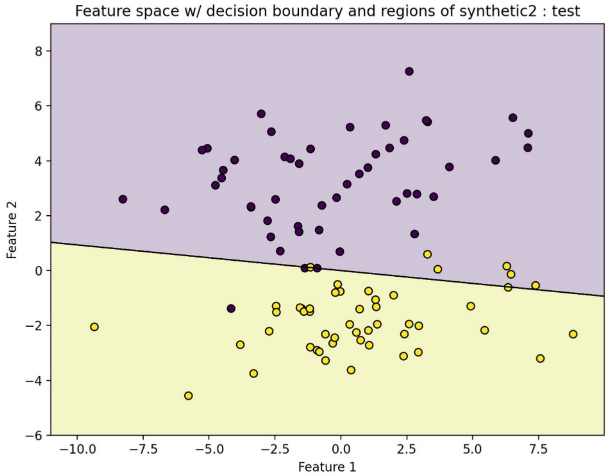
(b) The following are the results on the “synthetic2” dataset:

(i) The optimal weights $\underline{\hat{w}}$, i.e the global minima = **[0.1579255, 1.688338]**. The algorithm halted without convergence, with a final J($\underline{\hat{w}}$) = **0.46854**.

(ii) After running the perceptron classifier on the training set and the test set, we obtain classification errors of **0.03 and 0.08**, respectively.

(iii) The following two figures illustrate the feature space of the training and test data set of synthetic2. We clearly see that there are 3 misclassifications in the training set and 8 in the test set.





(c) The following are the results on the "wine" dataset:

(i) The optimal weights $\hat{\underline{w}}$, i.e the global minima = **[-3505.025, 2358.549, 592.752, -5012.76, -103.1, -2564.207, -1221.414, 106.042, -737.086, -231.54, -401.67, -142.318, 145.7].** The algorithm halted without convergence, with a final J($\hat{\underline{w}}$) = **113340.7303.**

(ii) After running the perceptron classifier on the training set and the test set, we obtain classification errors of **0.1538 and 0.2**, respectively.

(iii) **No, the data is not linearly seperable**. Projection of the wine data onto the feature space, would lead to a 13-dimensional space, which is difficult to analyse and to conclude if the data is linearly seperable. And so, this leads us to resorting to dimensionality reduction techniques like Principal Component Analysis (PCA) and t-distributed stochastic neighbour embedding (t-SNE), to model high-dim data.

3. The criterion function $J(\underline{w})$ of a multiclass perceptron is as follows:

$$J(\underline{w}) - \sum_{n=1}^{N} [[\, l_n \neq k_n]] \, [\, \underline{w}_{k_n}^T \underline{x}_n - \underline{w}_{l_n}^T \underline{x}_n]$$

where $g_{k_n}(x_n) \rightarrow$ discriminant of correct class

$g_{l_n}(x_n) \rightarrow$ discriminant of predicted class by the ML model

$$l_n = \underset{m \in \{1,2,\dots c\}}{\arg\max} \{ g_m(x_n) \}.$$

$c \rightarrow$ total classes

Then, choose the max discriminant function.

(a) The batch GD weight update formula is:

$$\underline{w}(i+1) = \underline{w}(i) - \eta \nabla_{\underline{w}} J(\underline{w}).$$

As above,

$$l_n = \underset{m \in \{1,2,\dots c\}, (m \neq k)}{\arg\max} \{ g_m(\underline{x}_n^{(k)}) \}$$

$c \rightarrow$ total classes.

If $k_n \neq l_n$, we will get two weight update formula: one for k_n & the other l_n:

$$\underline{w}_{k_n}^{(i+1)} = \underline{w}_{k_n}^{(i)} - \eta \nabla_{\underline{w}_{k_n}} J(\underline{w}).$$

$$= \underline{w}_{k_n}^{(i)} + \eta \underline{x}_n^{(k)} \quad //$$

$$\underline{w}_{l_n}^{(i+1)} = \underline{w}_{l_n}^{(i)} - \eta \nabla_{\underline{w}_{\arg\max_{m \in (1,2,\dots c),\, m \neq k} g_m(\underline{x}_n^{(k)})}} J(\underline{w})$$

Find gradients of all c weight vectors and take the maximum value for l_n computation.

$$\underline{w}_{l_n}^{(i+1)} = \underline{w}_{l_n}^{(i)} - \eta \underline{x}_n^{(k)} \quad //$$

If $k_n = l_n$, $\underline{w}_{k_n}(i+1) = \underline{w}_{k_n}(i) \quad //$

(b) <u>Sequential gradient descent</u> :

For each data point, weight updation is as follows :

$\underline{w}(i+1) = \underline{w}(i) - \eta \nabla_{\underline{w}} J_n(\underline{w}) \qquad \eta(i) \geq 0$.

From previous, if $k_n = l_n$ i.e $g_{k_n}(x_n) > g_{l_n}(x_n) \quad \forall j \neq k$,
there is no weight update
or $\underline{w}_{k_n}(i+1) = w_{k_n}(i) \qquad (\because J(\underline{w}) = 0)$.

If $k_n \neq l_n$, (incorrect classify)
$w_{k_n}(i+1) = w_{k_n}(i) + \eta x_n^{(k)} \qquad \eta(i) \geq 0$
$w_{l_n}(i+1) = w_{l_n}(i) - \eta x_n^{(k)} \qquad \eta(i) \geq 0$.

<u>Hence, the algorithm is as follows</u> :
Randomly shuffle the order of data points ..
Initialise $\underline{w}(0)$ i.e $\underline{w}_1(0) = \underline{w}_2(0) = \cdots = w_c(0)$ for C classes
Define seq-epoch m as :
for n in $\{1,2,\ldots N\}$
$i = (m-1)N + (n-1)$; $k_n \in \{1,2,\ldots,C\}$.
$l_n = \underset{m \in \{1,2,\ldots,C\}, m \neq k}{\operatorname{argmax}} \; g_m(\underline{x_n}^{(k)})$.
if $l_n \neq k_n$ (incorrect classify) $\quad k_n \rightarrow$ correct label from dataset .
$\underline{w}_{k_n}(i+1) = \underline{w_{k_n}(i) + \eta x_n^{(k)}}$
$\underline{w}_{l_n}(i+1) = \underline{w_{l_n}(i)} - \eta \underline{x_n}^{(k)}$
else
$\underline{w}_{k_n}(i+1) = \underline{w}_{k_n}(i) \quad \forall k_n \in \{1,2,\ldots,C\}$. //
until : all points are correctly classified .

(c) This weight-update in (b) is very similar to what was in lecture ~~class~~. However, we initialise each class of the weight vector at first, to compute the gradients (for each class). We make sure $J(\underline{w}) = \sum_{n=1}^{N} J_n(\underline{w})$.

4. A 2-class perceptron with a margin has the following criterion function :

$$J(\underline{w}) = -\sum_{n=1}^{N} [[\underline{w}^T z_n \underline{x_n} \leq b]] \; \underline{w}^T z_n \underline{x_n} \; ;$$ where b is a marginal value $\in \mathbb{R}$.

Clearly, there will be a misclassification when $\underline{w}^T z_n \underline{x_n} \leq b$.

We make few assumptions : fixed increment i.e $\eta(i) = \eta > 0$.

$$\nabla_{\underline{w}} J(\underline{w}) = -\sum_{n=1}^{N} [[\underline{w}^T z_n \underline{x_n} \leq b]] \; z_n \underline{x_n}$$ where $[[\cdot]] \rightarrow$ indicator function.

At first, we randomly shuffle the data $\rightarrow$ as per the sequential gradient approach.

Initialise $\underline{w}(0) \rightarrow$ init. weight vector :

$\underline{w}(0) \rightarrow$ arbitrary.

$$\underline{w}(i+1) = \underline{w}(i) + \eta \sum_{n=1}^{N} [[\underline{w}^T z_n \underline{x_n} \leq b]] \; z_n \underline{x_n}.$$

or $\underline{w}(i+1) = \underline{w}(i) + \eta [[\underline{w}^T z_i \underline{x_i} \leq b]] \; z_i x_i$

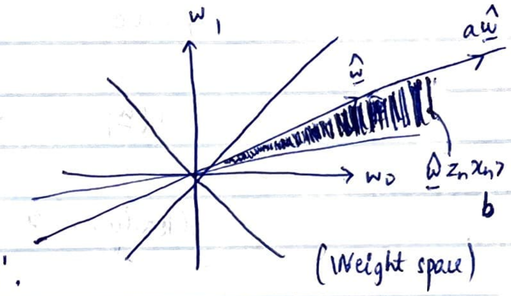
$z_i x_i \rightarrow$ cyclically ordered training data points (over a series of epochs).

At the i^{th} iteration, let the point $z^i x^i$ be misclassified where $z^i x^i \rightarrow$ reflection of the data point x^i. or $\underline{w}(i)^T z^i x^i \leq b$.

and so, the corresponding weight update :

$w(i+1) = w(i) + \eta z^i x^i.$; $\eta > 0$ where $w(i)^T z^i x^i - b \leq 0 . \forall i$

Let $\hat{w}$ be an optimum weight vector solution i.e



$\hat{w}^T z_n x_n \geq b \quad \forall n$.

then, $a\hat{w}$ be a solution as well.

$a\hat{w}^T z_n x_n > b \quad \forall n, \quad a > 0$

Note : The solution region lies within 'b'.

Error : $\mathcal{E}_w(i) = \| \hat{w}(i) - a\hat{w}(i) \|_2^2$. where a is adjustable.

Hence, the error must decrease at each iteration and this what we choose to prove.

$\rightarrow \quad w(i+1) - a\hat{w} = (w(i) - a\hat{w}) + \eta z^i x^i, \quad a > 0$

Taking squared of L2 norm,

or $\| w(i+1) - a\hat{w} \|_2^2 = \| w(i) - a\hat{w} \|_2^2 + \| \eta z^i x^i \|_2^2 + 2 [w(i) - a\hat{w}]^T \eta z^i x^i, \quad a > 0$

Since the increment is fixed, let's drop η.

~~Now, $w(i)\, x^i$~~

$w(i)\, z^i x^i \leq b$.

So, $\| w(i+1) - a\hat{w} \|_2^2 \leq \| w(i) - a\hat{w} \|_2^2 + \| z^i x^i \|_2^2 - 2a\hat{w}^T z^i x^i + 2b$.

but since $\hat{w}$ is optimal weight vector, $\hat{w}^T z^i x^i \geq b$.

Let $K^2 = \max_j \| x_j \|_2^2$ = {Length of longest x-vector}^{2}

and $l \triangleq \min_j \{ \hat{w}^T z^i x^i \} > b$.

$\therefore \quad \| w(i+1) - a\hat{w} \|_2^2 \leq \| w(i) - a\hat{w} \|_2^2 + K^2 - 2al + 2b$.

Let $a = \dfrac{K^2}{l}$.

$\therefore \quad \| w(i+1) - a\hat{w} \|_2^2 \leq \| w(i) - a\hat{w} \|_2^2 - K^2 + 2b$.

or $\mathcal{E}_w(i+1) - \mathcal{E}_w(i) \leq -K^2 + 2b$.

So, the error reduces by $-K^2 + 2b$ when b is the margin.

or $\mathcal{E}_{\underline{w}}(i+1) - \mathcal{E}_{\underline{w}}(i) \leq -k^2 + b$ (approx).

Apply forcing argument :

$0 \leq \mathcal{E}_{\underline{w}}(i+1) \leq \mathcal{E}_{\underline{w}}(i) - k^2 + b \leq 0$

For some i_0, we'd have,

$0 \leq \mathcal{E}_{\underline{w}}(i_0+1) \leq \mathcal{E}_{\underline{w}}(i_0) - k^2 + b < 0$

$\longrightarrow$ impossible

Iterations must cease at $i_0 - 1$ or sooner //

Hence, algorithm converges at a solution weight vector at $(i_0 - 1)^{th}$ iteration or sooner.

The 2-class perceptron with margin converges for linearly seperable data (training) //