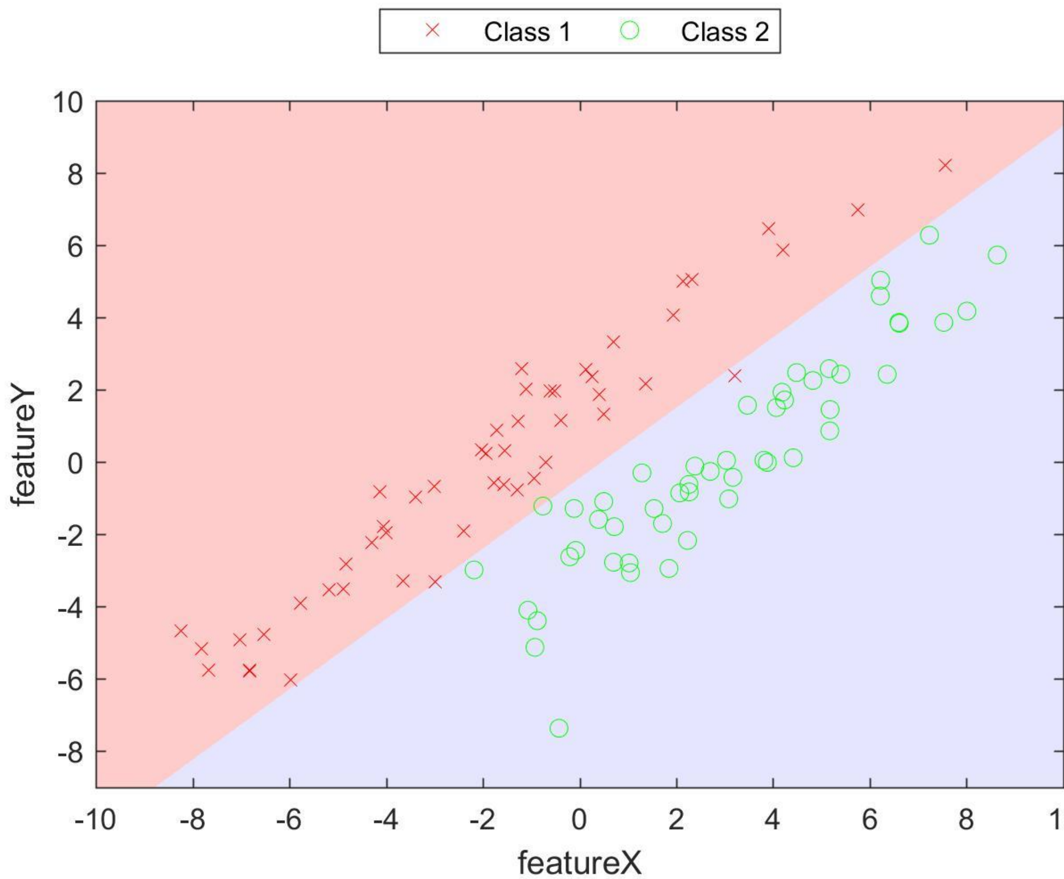
Q1

Synthetic 1 dataset

Final Weight Vector: 35.1000 -43.9945 45.6809

Train_error : 0.01

Test_error: 0.03



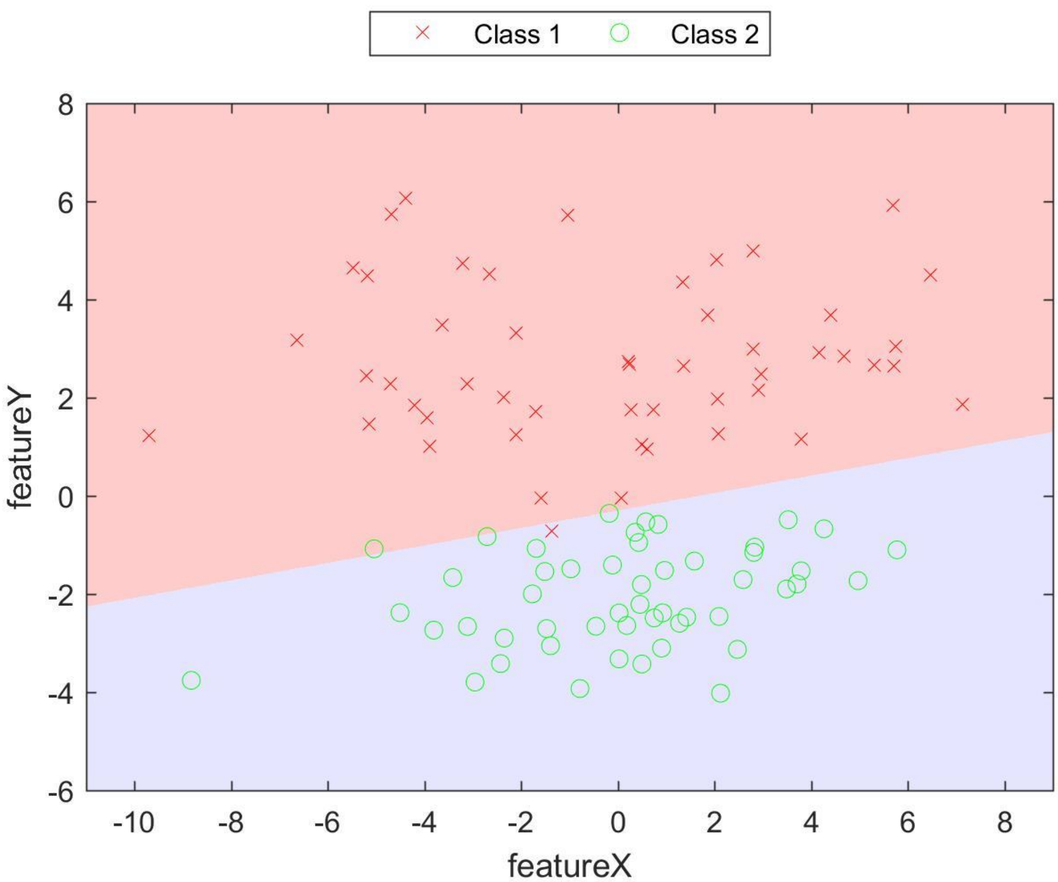
The error for training data and test data was 0.21 and 0.24 respectively in HW2(a). In this case, the classifier works much better and gives a much lesser error. Thus we can say that the perceptron is a much better classifier algorithm than the MDTCM algorithm.

Synthetic 2 dataset

Final Weight Vector: 6.1000 -3.6526 20.5332

Train_error : 0.02

Test_error: 0.03



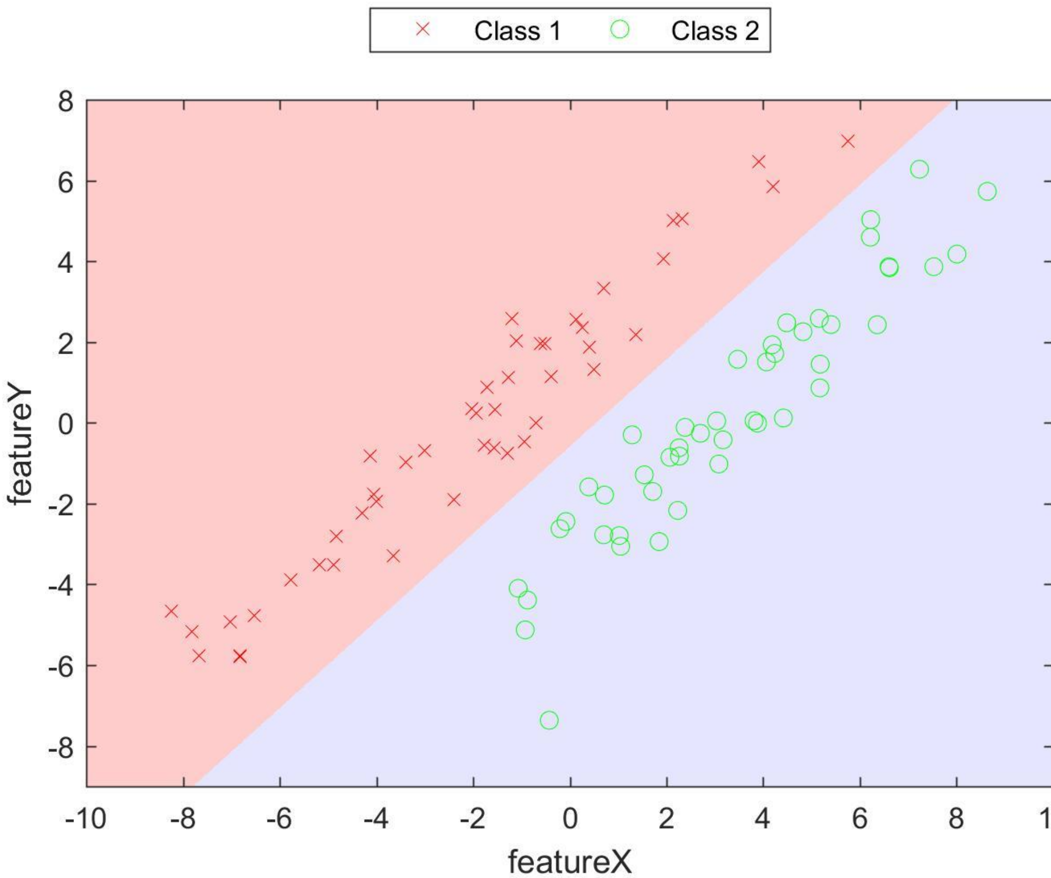
The error for training data and test data was 0.03 and 0.04 respectively in HW2(a). In this case, the classifier works better and gives a lesser error. Thus we can say that the perceptron is a much better classifier algorithm than the MDTCM algorithm.

Synthetic 3 dataset

Final Weight Vector: 0.1000 -6.9558 6.0228

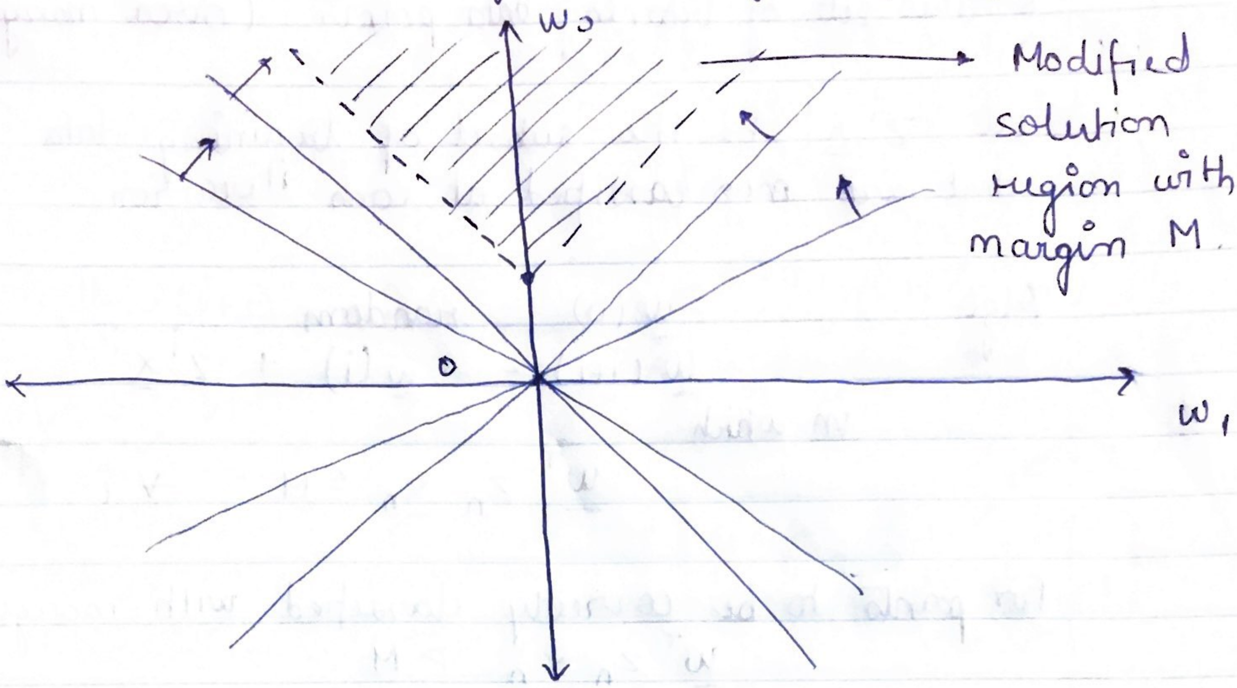
Train_error : 0

Test_error: 0.01



The perceptron algorithm gives a reduced error rate, thus showing that it can better adjust the weight vectors for the features, resulting in better decision boundary.

Q.2 Perceptron with margin convergence proof



ASSUMPTIONS:

- Fixed Increments $\eta(i) = \eta = \text{constant} > 0$
- Sequential Gradient Descent
- Data points are linearly separable
- Use reflected data points $z_n \underline{x}_n$, $n = 1, 2, \cdots, N$

We can set $\eta = 1$, without loss of generality:
let $z_n \underline{x}_n' = z_n \underline{x}_n$, $\eta > 0$
Then drop primes.

ALGORITHM:

- $\underline{w}(0) = \text{arbitrary}$
- $\underline{w}(i+1) = \underline{w}(i) + z_i x_i \quad [\![\underline{w}^T z_i \underline{x}_i \leq M]\!]$

in which $z_i \underline{x}_i$, $i = 1, 2, \cdots$ are the cyclically ordered set of training data points (over many epochs).

Let $z^i \underline{x}^i$ be the subset of training data points that are misclassified at each iteration.

Algo:
$$\underline{w}(0) = \text{random}$$
$$\underline{w}(i+1) = \underline{w}(i) + z^i \underline{x}^i$$

in which
$$\underline{w}^T z_n \underline{x}_n \leq M \quad \forall i$$

For points to be correctly classified with margin M,
$$\underline{w}^T z_n \underline{x}_n > M$$

If $\hat{\underline{w}}$ is a solution, then $a\hat{\underline{w}}$, $a > 1$ is also a solution

$$\therefore \quad \hat{\underline{w}}^T z_n \underline{x}_n > M \quad \forall n$$

$$\Rightarrow a\, \hat{\underline{w}}^T z_n \underline{x}_n > aM \quad \forall n$$

- Need 'error measure' on $\underline{w}(i)$:

$$E_{\underline{w}}(i) = \| \underline{w}(i) - a\hat{\underline{w}} \|_2^2$$

- Show $E_{\underline{w}}(i)$ must decrease at each iteration $\Rightarrow$

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

$$\Rightarrow \| \underline{w}(i+1) - a\hat{\underline{w}} \|_2^2 = \| \underline{w}(i) - a\hat{\underline{w}} \|_2^2 + 2 [\underline{w}(i) - a\hat{\underline{w}}]^T z^i \underline{x}^i + \| z^i \underline{x}^i \|_2^2$$

In this, $\quad 2\,\underline{w}^{(i)T} z^i \underline{x}^i \leq 2M$

$$\Rightarrow \|w(i+1) - a\hat{\underline{w}}\|_2^2 \leq \|w(i) - a\hat{\underline{w}}\|_2^2 - \underbrace{2a\hat{w}^T z^i \underline{x}^i}_{> aM} - 2\underline{w}(i)^T z^i \underline{x}^i + \|z^i \underline{x}^i\|_2^2$$

$$\Rightarrow \|w(i+1) - a\hat{\underline{w}}\|_2^2 \leq \|w(i) - a\hat{\underline{w}}\|_2^2 + 2M - 2aM + \underbrace{\|z^i \underline{x}^i\|_2^2}_{>0}$$

Let $\quad b^2 \triangleq \max_j \|x_j\|_2^2 = \left[\text{length of largest data point (vector)}\right]^2$

$$\therefore \|w(i+1) - a\hat{\underline{w}}\|_2^2 \leq \|w(i) - a\hat{\underline{w}}\|_2^2 + 2M - 2aM + b^2$$

Now choose $\quad a = \dfrac{b^2 + M}{M} > 1$, we get,

$$\|w(i+1) - a\hat{\underline{w}}\|_2^2 \leq \|w(i) - a\hat{\underline{w}}\|_2^2 - b^2$$

$$\Rightarrow \quad E\,\underline{w}(i+1) \leq E_{\underline{w}}(i) - b^2$$

$\Rightarrow$ so each iteration reduces $E_{\underline{w}}$ by at least b^2.

- Applying forcing argument

$$0 \leq E_{\underline{w}}(i+1) \leq E_{\underline{w}}(i) - b^2 \qquad \forall\, i$$

For some i_0, we would have $E\,\underline{w}(i_0) < b^2$
so that,

$$0 \leq E\,\underline{w}(i+1) \leq E\,\underline{w}(i) - b^2 < 0$$

which is impossible

$\Rightarrow$ Iteration must cease at $i = i_0$ (or sooner)

$\therefore$ Algorithm converges at a solution weight vector at $(i_0 - 1)^{th}$ iteration or sooner.

Q3. a) $\Delta \underline{w}(i) = \underline{w}(i+1) - \underline{w}(i)$

$J(\underline{w}) = \sum_{n=1}^{N} J_n(\underline{w})$

$\eta(i) = \eta$

$\therefore E\{\Delta \underline{w}(i)\} = E\{\underline{w}(i+1) - \underline{w}(i)\}$

$= E\{\underline{w}(i+1)\} - E\{\underline{w}(i)\}$

$= E\{\underline{w}(i) + \eta z_n \underline{x}_n^{(i)}\} - E\{\underline{w}(i)\}$

$= E\{\underline{w}(i)\} + E\{\eta z_n \underline{x}_n^{(i)}\} - E\{\underline{w}(i)\}$

$= \eta E\{z_n \underline{x}_n^{(i)}\} = \eta \sum z_n \underline{x}_n^{(i)} \cdot p(z_n \underline{x}_n^{(i)})$

$= -\frac{\eta}{N} \nabla_w J(\underline{w}) = -\frac{\eta}{N} \sum_{n=1}^{N} \nabla_w J_n(w)$

b) $E\{\sum_{i=0}^{N-1} \Delta \underline{w}(i)\}$ $\Delta \underline{w}(i)$ are iid

$= E\{N(\Delta \underline{w}(i))\}$

$= N \, E\{\Delta \underline{w}(i)\}$

$= -\eta \cdot \frac{N}{N} \cdot \nabla_w J(w)$ from (a)

$= -\eta \nabla_w J(\underline{w})$

$= -\eta \sum_{n=1}^{N} \nabla_w J_n(\underline{w})$

c) $$\Delta \underline{w}(i) = \eta \sum_{\substack{\text{all } n \text{ s.t.} \\ x_n \in \mathcal{X}}} z_n x_n$$

$$= -\eta \nabla_w J(\underline{w})$$

$$= -\eta \sum_{n=1}^{N} \nabla_w J_n(\underline{w}) .$$

Thus comparing (b) and (c), we can see that the expected value of the sum of weight difference for stochastic gradient descent, variant 2 is equal to the difference in weights for batch gradient descent.

In Stochastic Gradient Descent - Variant 2, we randomly pick a training data point (with replacement) and perform single sample update on the weight, whereas in block gradient descent, we update weight for all misclassified datapoints in one iteration.

Batch gradient descent is great for convex, or relatively smooth error manifolds. Whereas, stochastic gradient descent is better for error manifolds that have lots of local maxima / minima. Computationally, stochastic gradient descent is much faster. ~~[illegible]~~ This computational ~~[illegible]~~ advantage is leveraged by performing many more steps th an conventional batch gradient descent, resulting in a close model to that found by the latter.