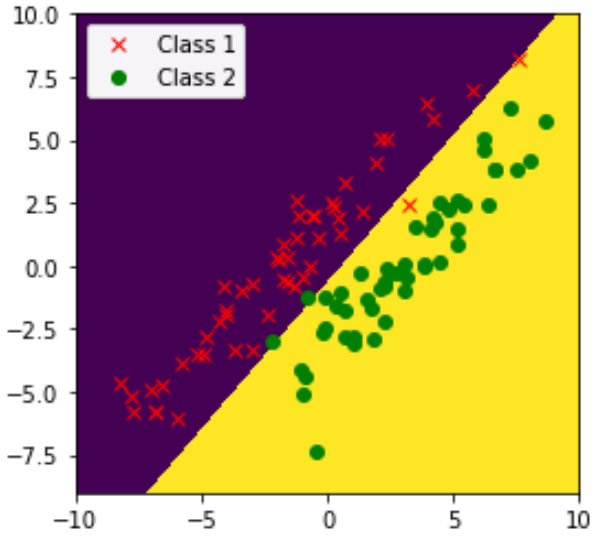
Problem 1
(a)
Final w on synthetic1 = [33.1 -65.44535 56.18591]
Error rate on synthetic1 training set = 4.0%
Error rate on synthetic1 test set = 0.0%

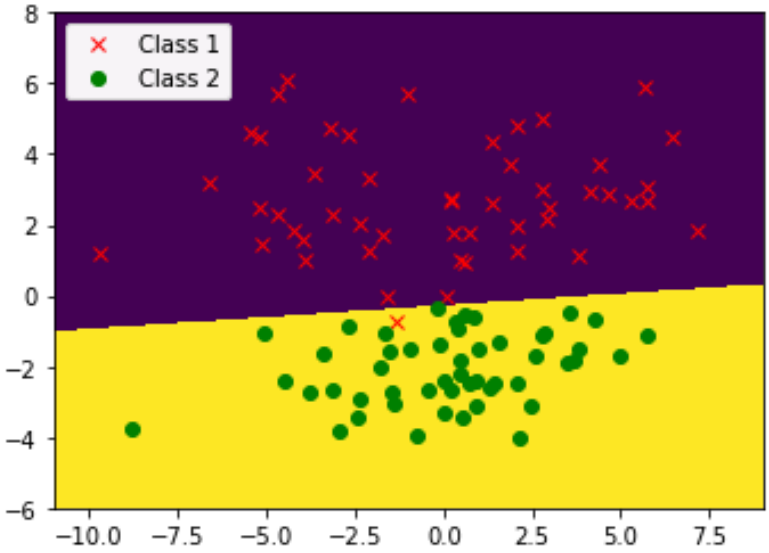
Final w on synthetic2 = [5.1 -1.25571 18.866108]
Error rate on synthetic2 training set = 1.0%
Error rate on synthetic2 test set = 5.0%

Final w on synthetic3 = [3.1 -22.37258 18.6587]
Error rate on synthetic3 training set = 0.0%
Error rate on synthetic3 test set = 0.0%

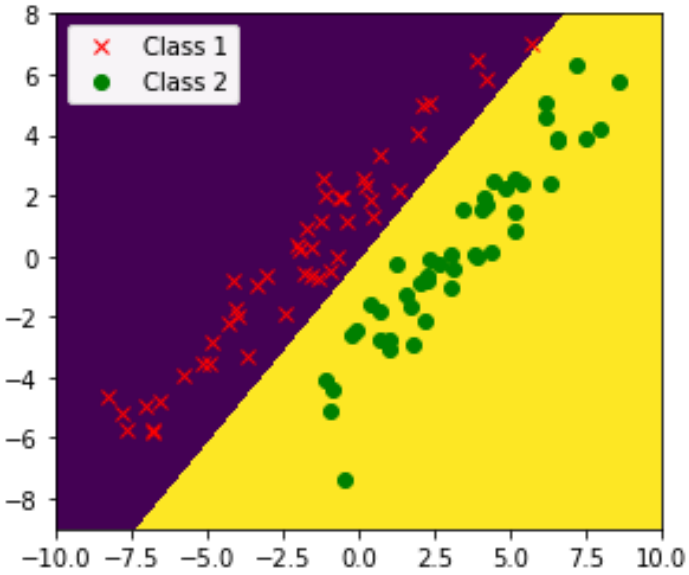
(b)
For synthetic1:



For synthetic2:



For sunthetic3:



(c)
In homework1:
Error rate of synthetic1 training set = 21.0%
Error rate of synthetic1 test set = 24.0%
The outcome of homework4 is much better than that of homework1, because the distribution of the two class in synthetic1 are oval with its long axis is much longer than short axis, and the two classes are close to each other by short axis, which leads to the point in one class may be closer to the other sample mean point than its own sample mean point, so the error rates are high. In perceptron learning, w is adjusted if there are any miss-classification point to make error rate lower, and the two classes are almost linear separable, so it could get to the decision boundary better than nearest mean method.

In homework1:
Error rate of synthetic2 training set = 3.0%
Error rate of synthetic2 test set = 4.0%
The outcome of homework4 is similar to homework1, error rate on training set is lower in homework4 and that on test set is higher in homework4. For the distribution of synthetic2 is more like a circle, which makes synthetic2 is suitable to nearest mean method, and the two classes fall apart enough. As for perceptron learning, it could find a the decision boundary for the two classes are almost linear separable.

#

2.(a). $E\{\Delta \underline{w}(i)\} = E\{\underline{w}(i+1) - \underline{w}(i)\} = E\{w_0 - \eta \nabla_{\underline{w}} J_n(w_0) - w_0\} = -\eta E\{\nabla_{\underline{w}} J_n(w_0)\}$. $n = 1 : N$

$P(\nabla_{\underline{w}} J_n(w_0)) = \frac{1}{n}$, So. $E\{\nabla_{\underline{w}} J_n(w_0)\} = \sum_{n=1}^{N} \frac{1}{n} \nabla_{\underline{w}} J_n(w_0) = \frac{1}{n} \sum_{n=1}^{N} \nabla_{\underline{w}} J_n(w_0) = \frac{1}{n} \nabla_{\underline{w}} J(w_0)$

~~2(b). Δw_i for batch gradient descent is ∇~~

So. $E\{\Delta \underline{w}(i)\}$ for stochastic gradient descent is $-\eta \frac{1}{n} \nabla_{\underline{w}} J(\underline{w}_0)$

(b). Δw_i for batch gradient descent is $-\eta \nabla_{\underline{w}} J(\underline{w}_0)$, Δw_i for batch gradient descent is n times in value. in comparasion with ~~stochastic~~ expected value of ~~so~~ stochastic GD variant 2 update.

it shows stochastic GD variant 2 will get the same expected result ~~as~~ in comparasion with batch GD which uses the information of the whole data set, even if stochastic GD. variant 2 only use single. data point to update weight. Besides, for the equivalent learn rate for stochastic GD variant 2. is $\frac{1}{n}$ of that of batch GD, so stochastic GD variant 2. could get a better result in some cases if the number of epoch is enough.