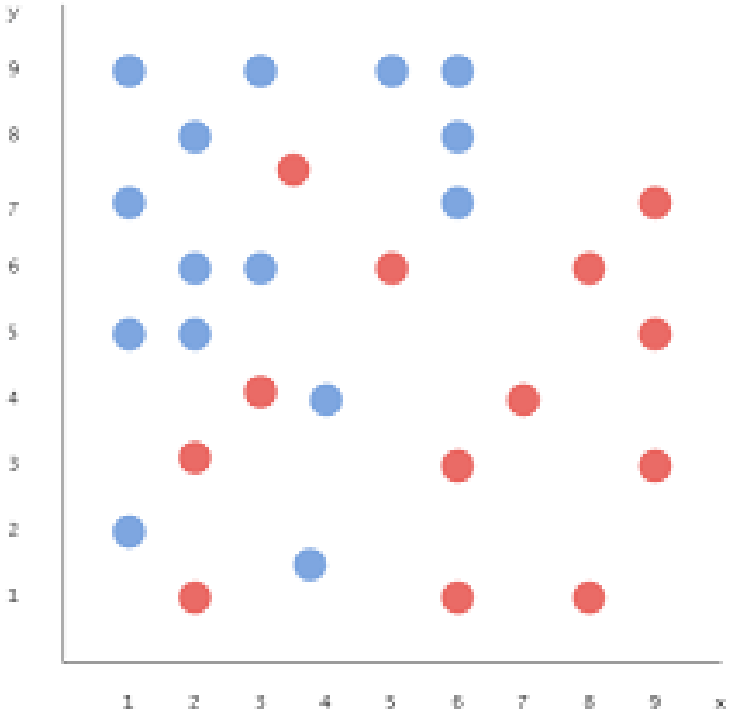
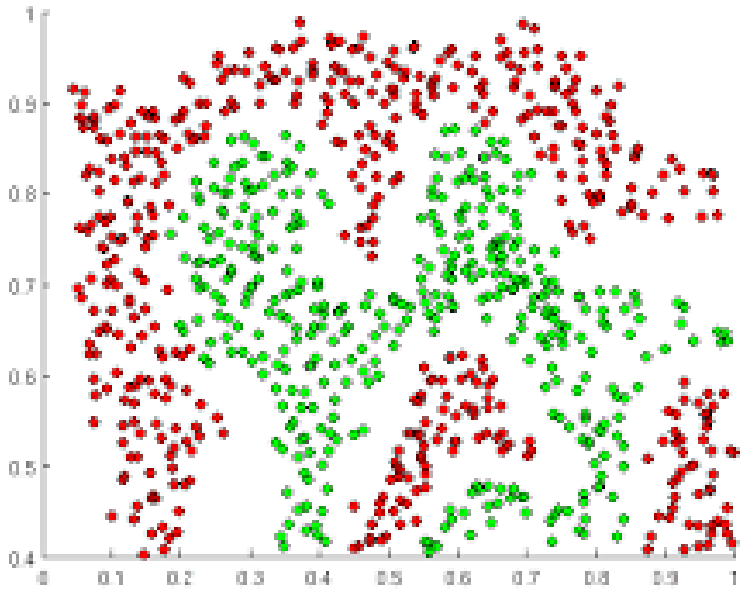
PERCEPTRON: PROOF OF CONVERGENCE

Dr. Umarani Jayaraman
Assistant Professor

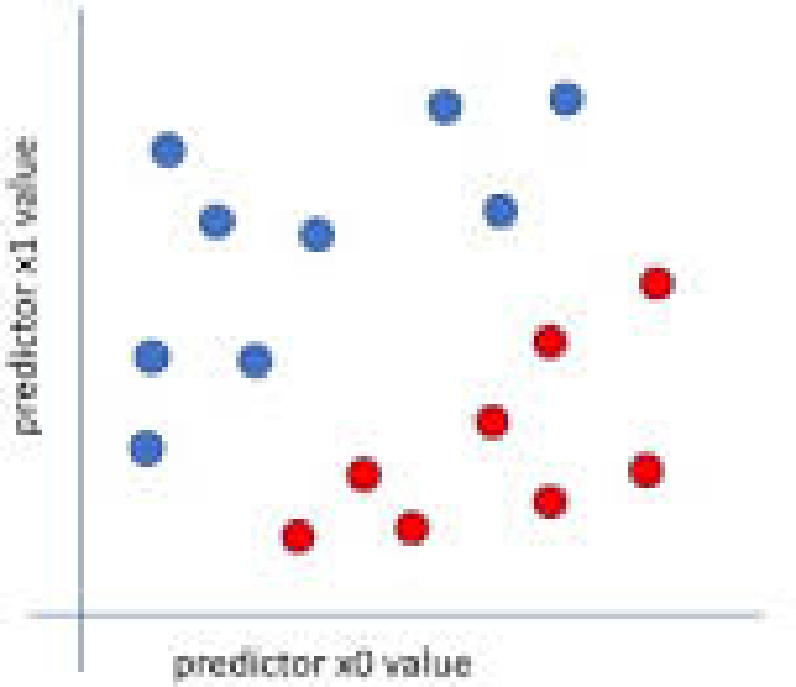
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY,
DESIGN AND MANUFACTURING,
KANCHEEPURAM

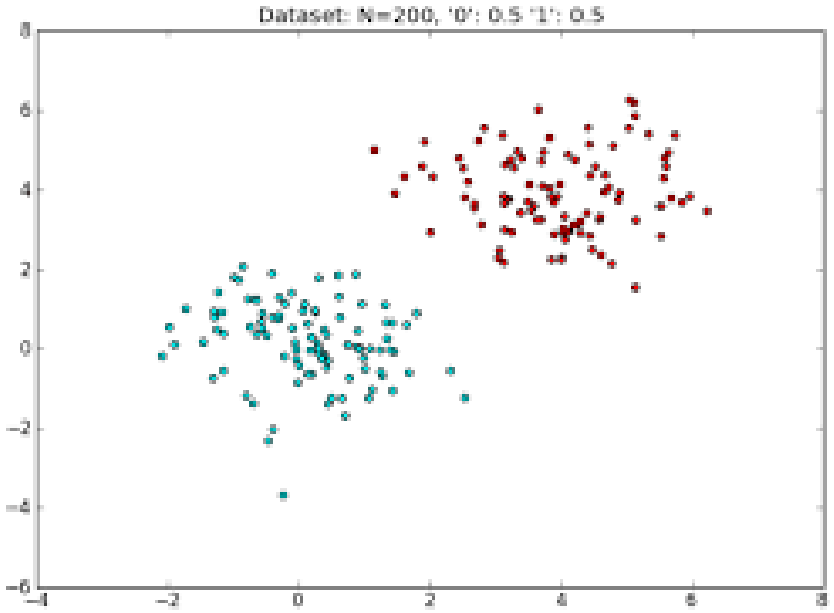
What classifier do you use?





What classifier do you use?

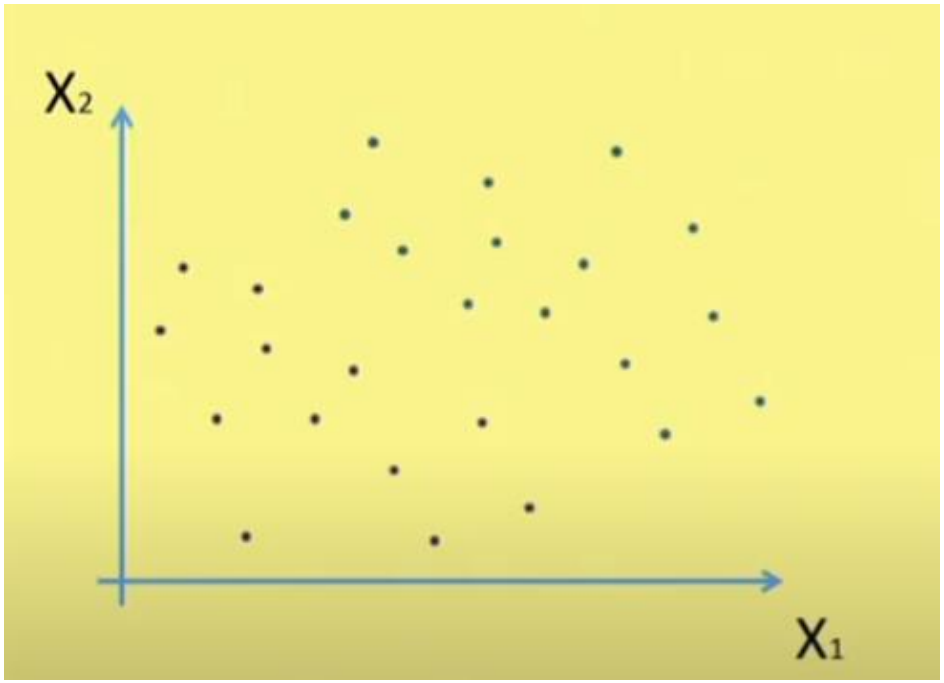




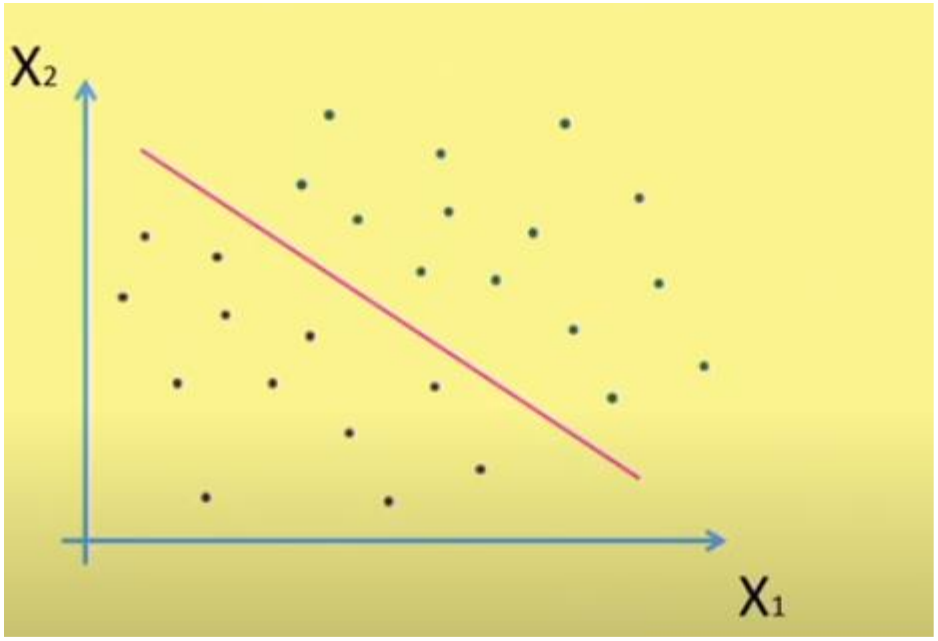
Possible solution for previous problems

- K-NN search
  - Testing: It is time consuming as it should compare the distance with all the sample
  - This methods works well if the number of samples are less
- Linear discriminant function (LDF)
  - It uses sample to estimate the parameter such as **w** and $\mathbf{w_0}$
  - Testing: It just substitute the value of x in g(x) and take the decision
  - If g(x) > 0 then class 1, otherwise class 2

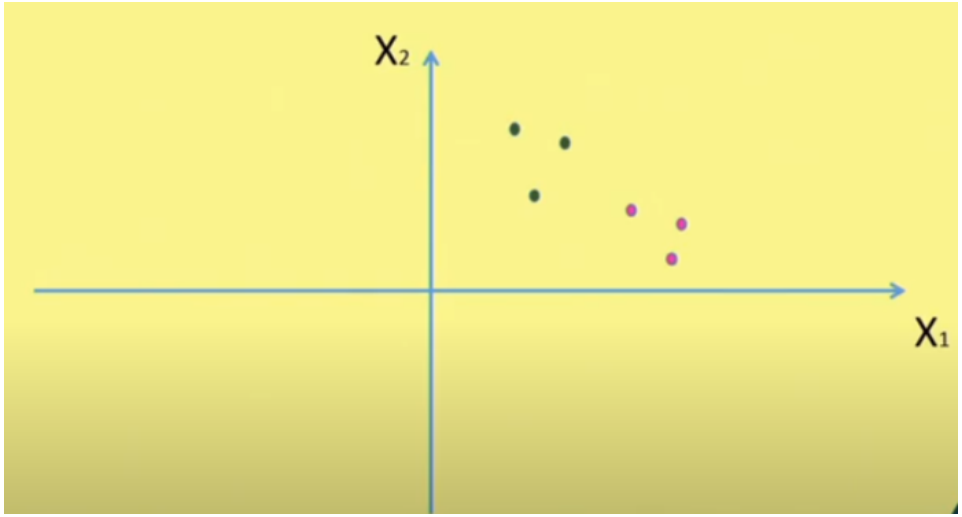
Perceptron: two class problem



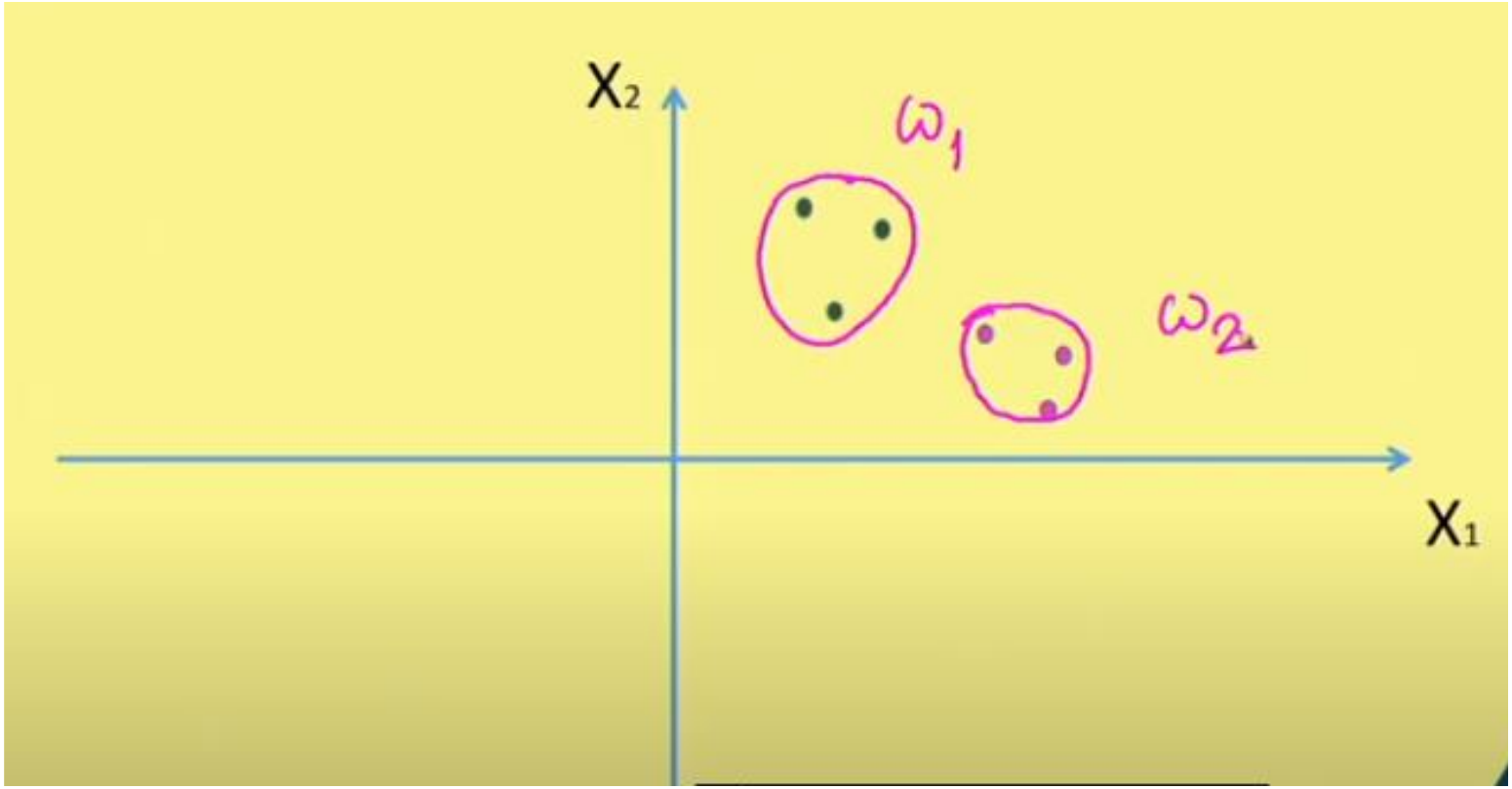
Perceptron - two class problem



Perceptron -two class problem

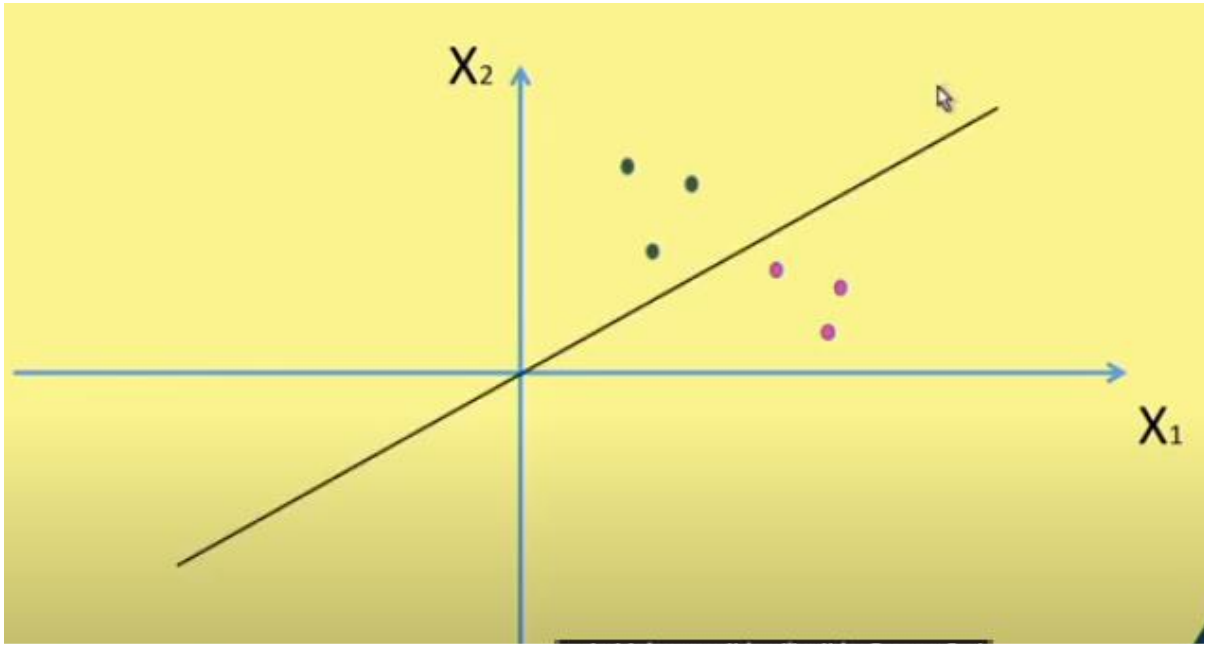


Linear Classifier: Perceptron



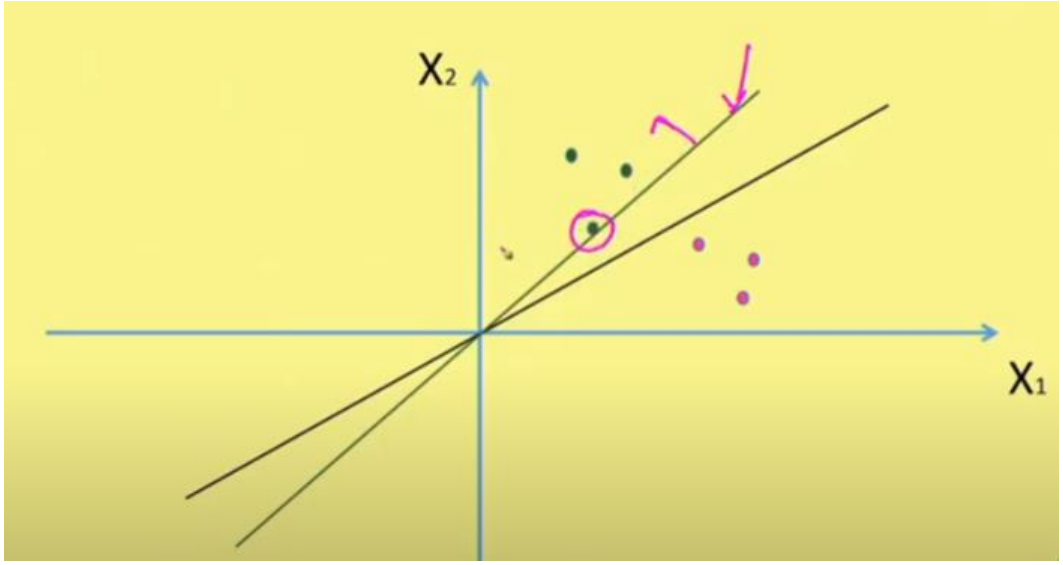
Linear Classifier: Perceptron

- This linear line clearly separates the two classes
- Also, we can see there is a limit of orientation of this particular line (decision boundary)



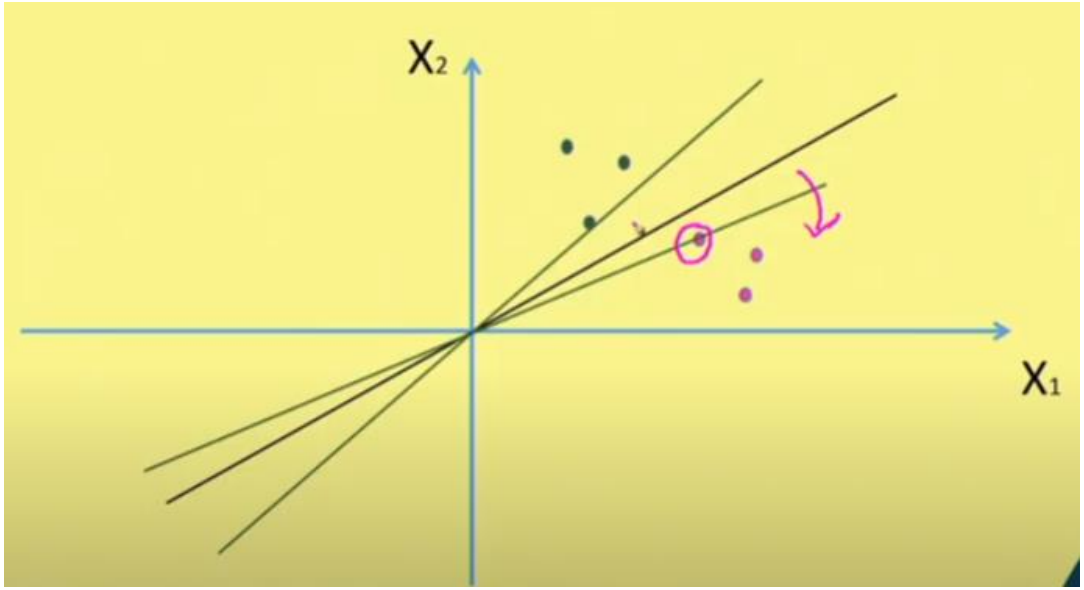
Linear Classifier: Perceptron

- If we rotate further in anti-clock wise direction, the misclassified sample is the one marked as circle.



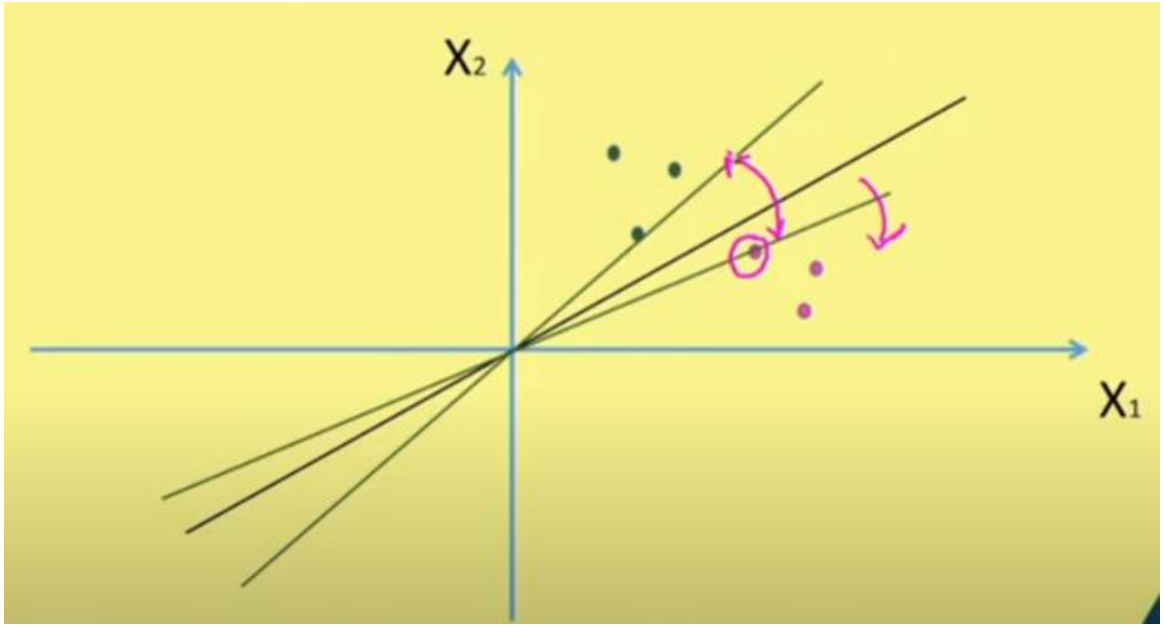
Linear Classifier: Perceptron

- Similarly, if we come to the other side, this is the limit
- If we rotate it further in clock wise direction; the misclassified sample is shown in circle



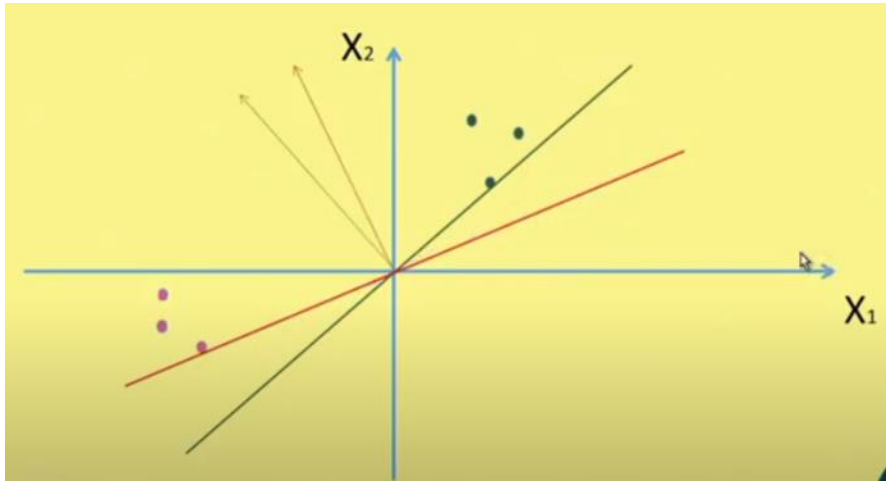
Linear Classifier: Perceptron

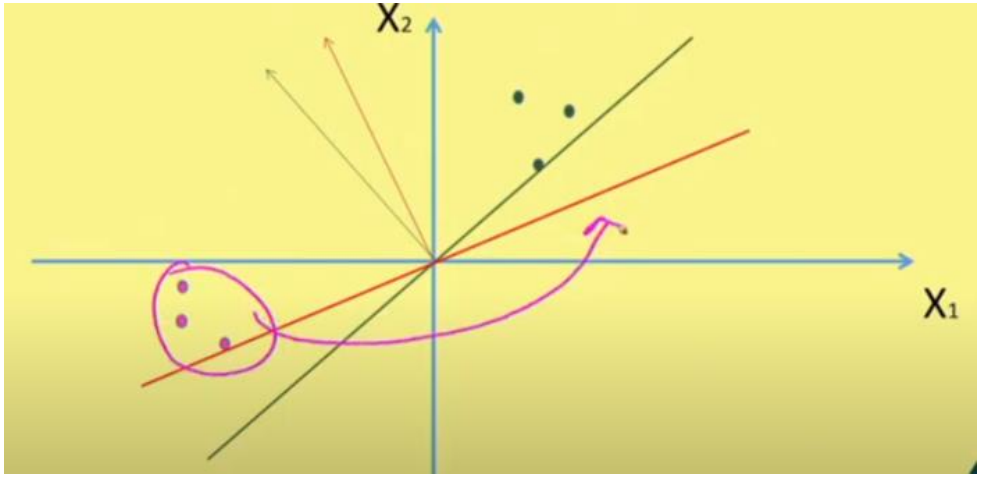
- So the range, that we could have is shown here.



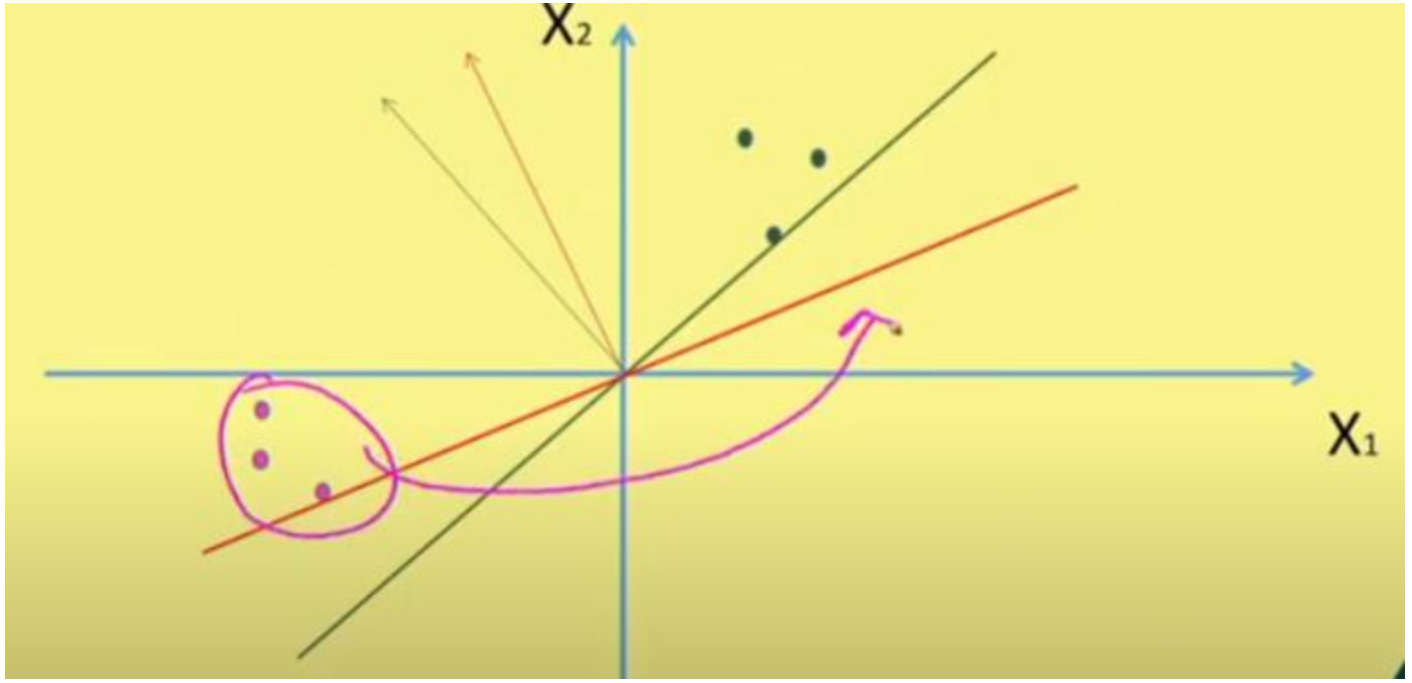
Linear Classifier: Perceptron

- After taking negation of all sample for class 2



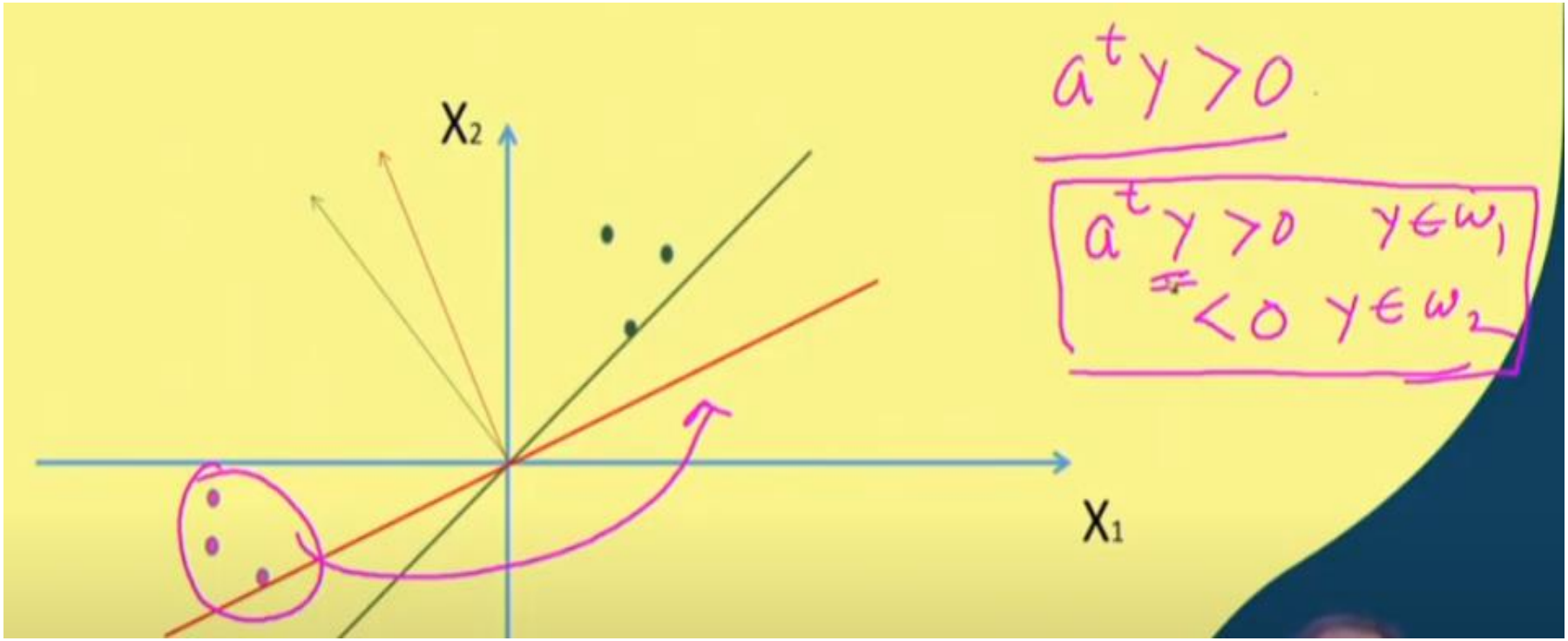


Linear Classifier: Perceptron

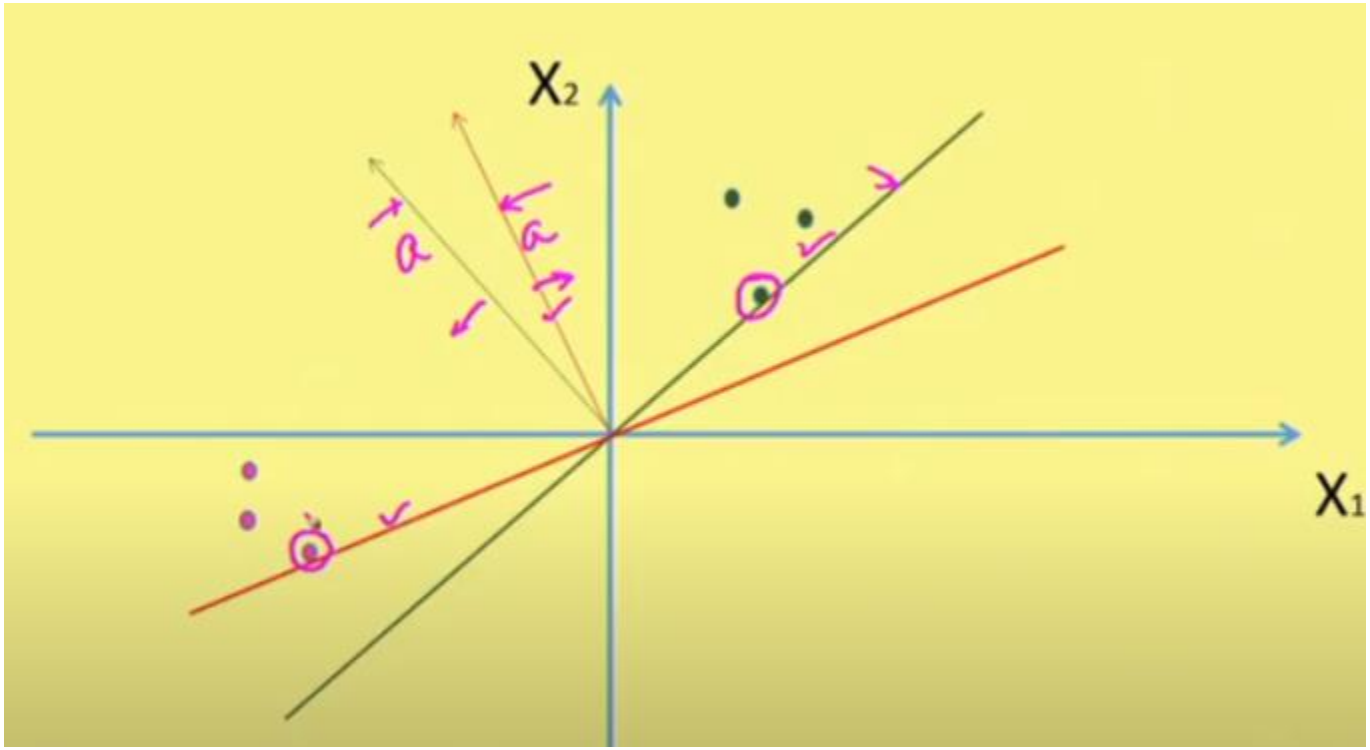


Linear Classifier: Perceptron

- For design purpose alone we use $a^t y > 0$
- For testing we use if $a^t y > 0$, then y is class 1
- If if $a^t y < 0$, then y is class 2

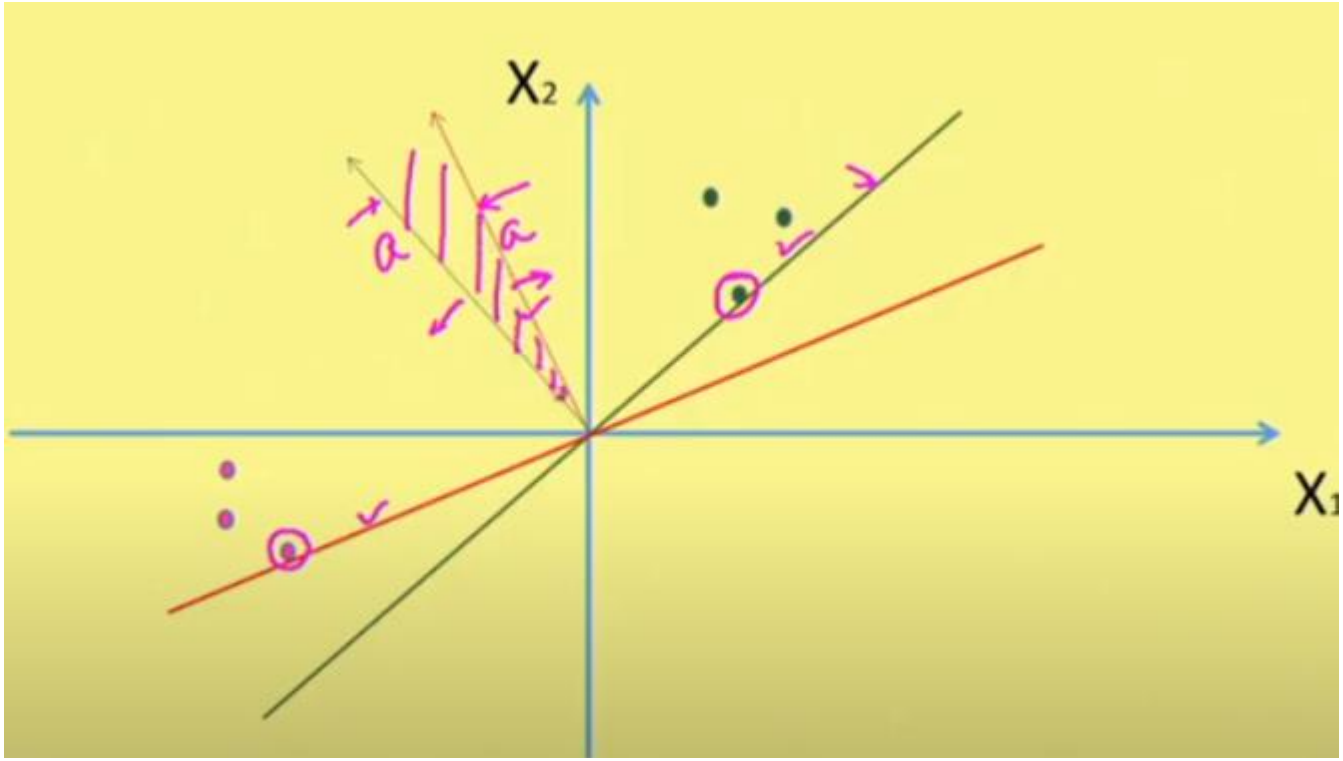


Linear Classifier: Perceptron



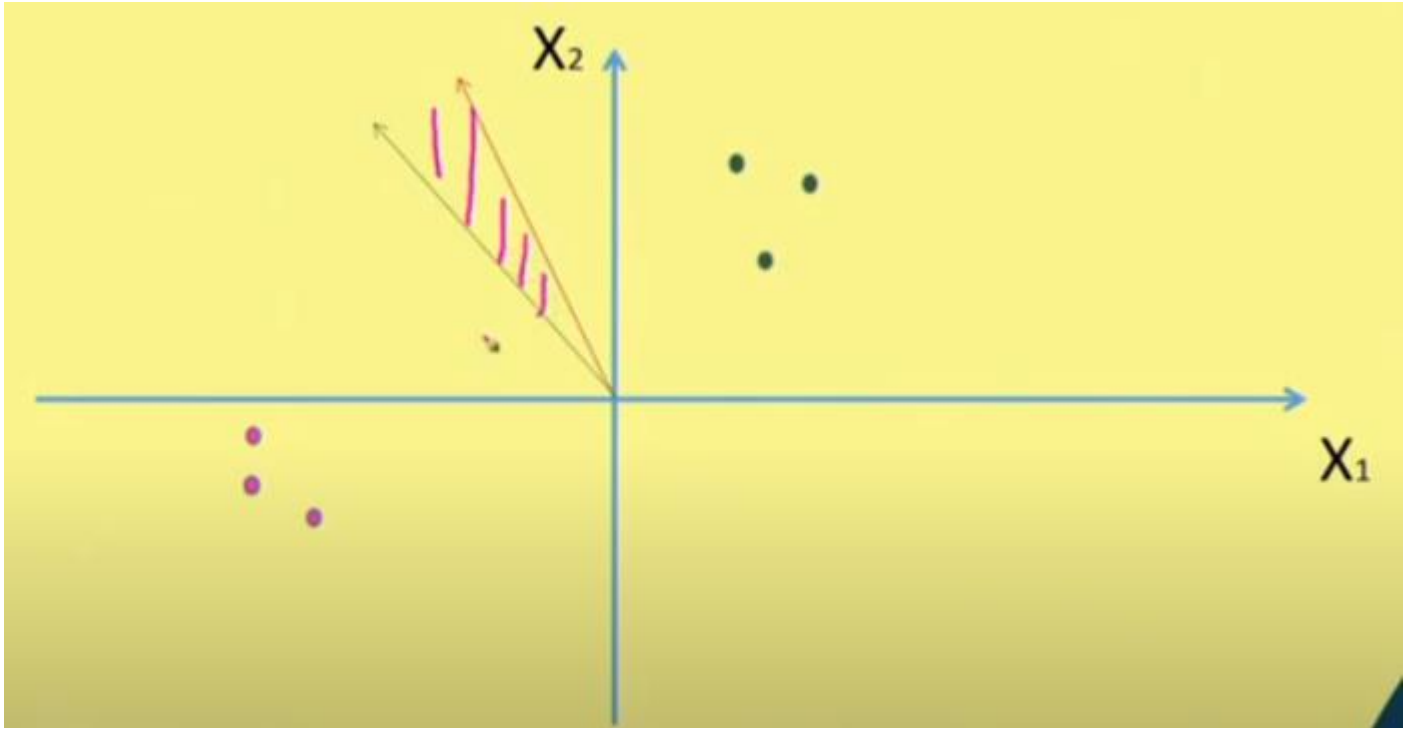
Linear Classifier: Perceptron

- So the solution vector ‘a’ must lie on this solution region.



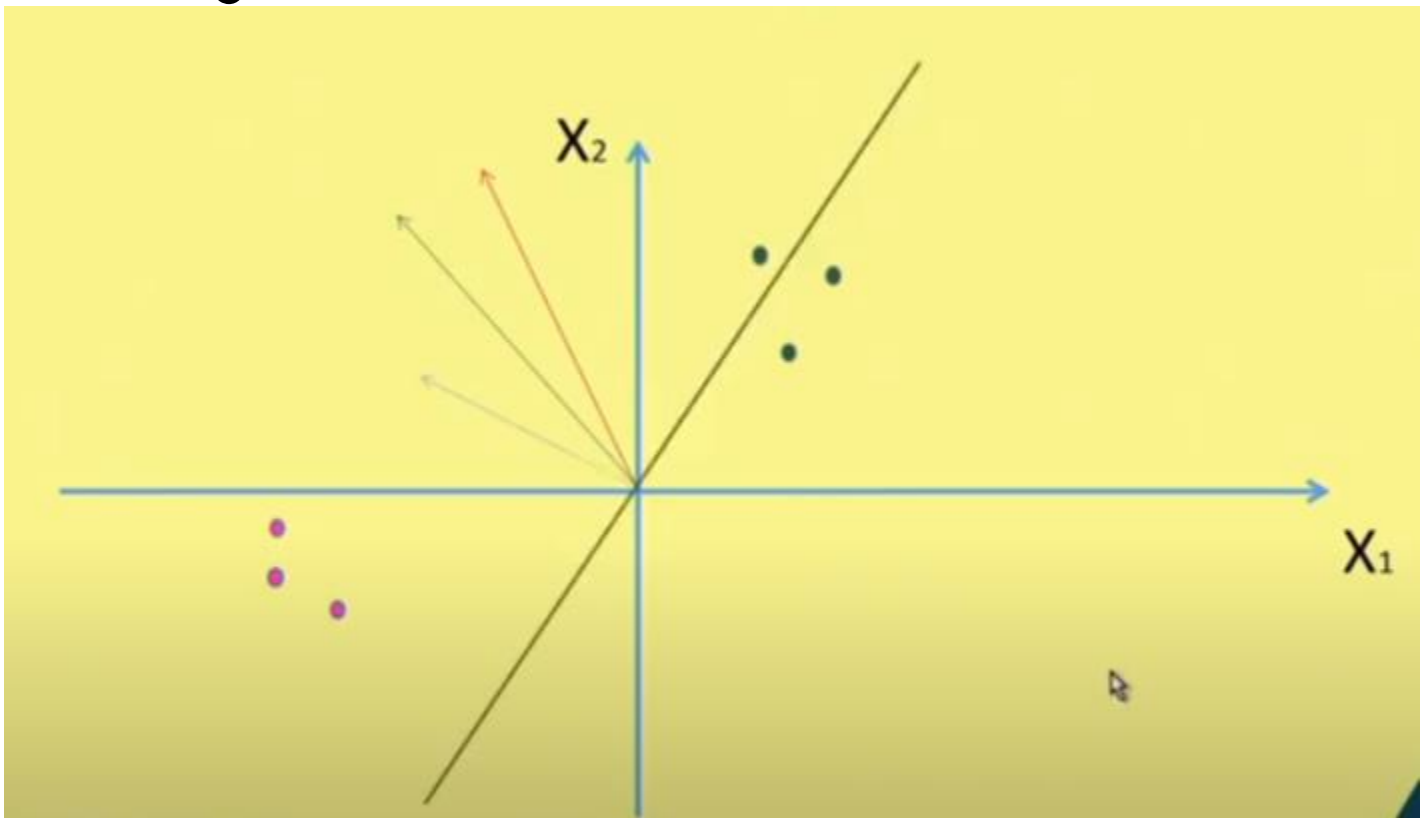
Linear Classifier: Perceptron

- How do we make the solution vector ‘a’ should land in solution region?

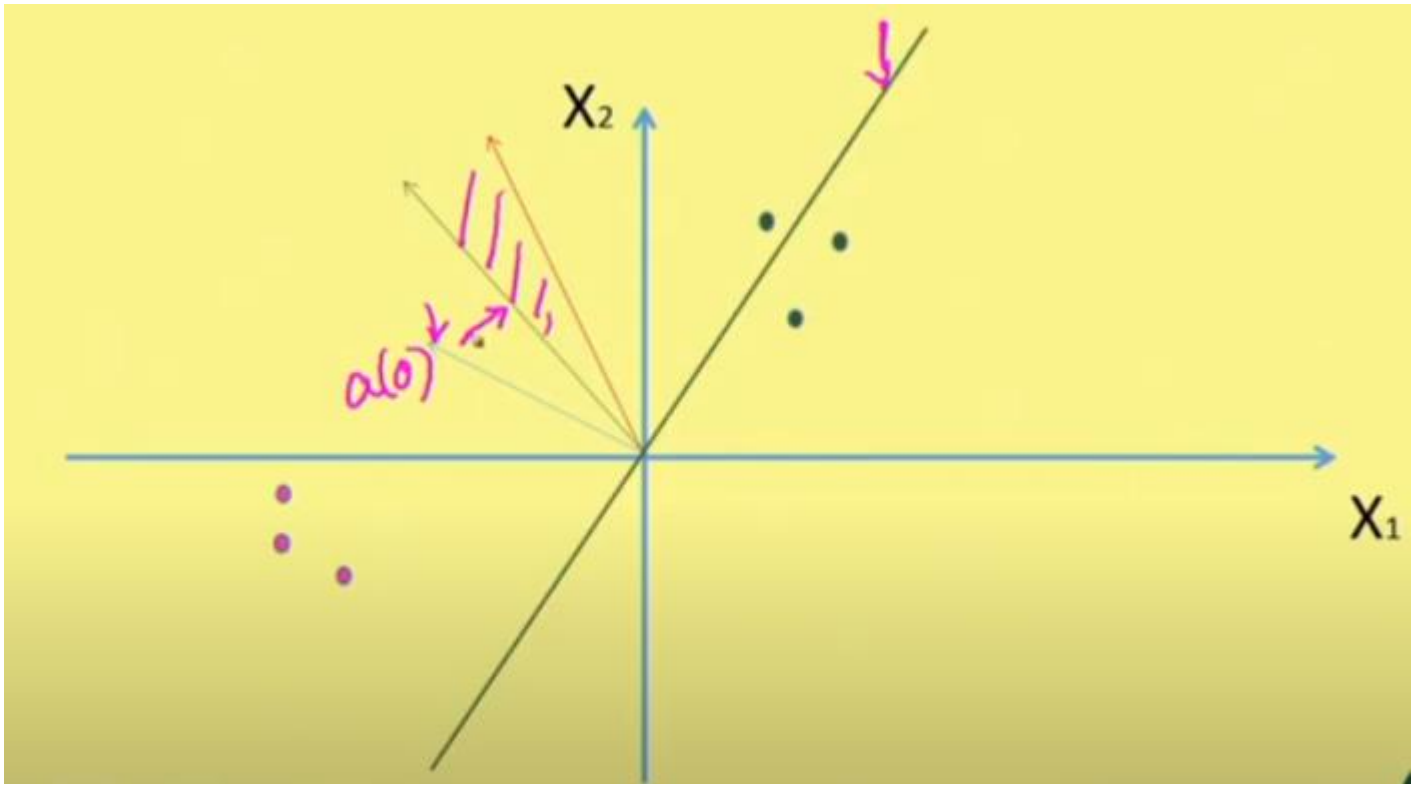


Linear Classifier: Perceptron

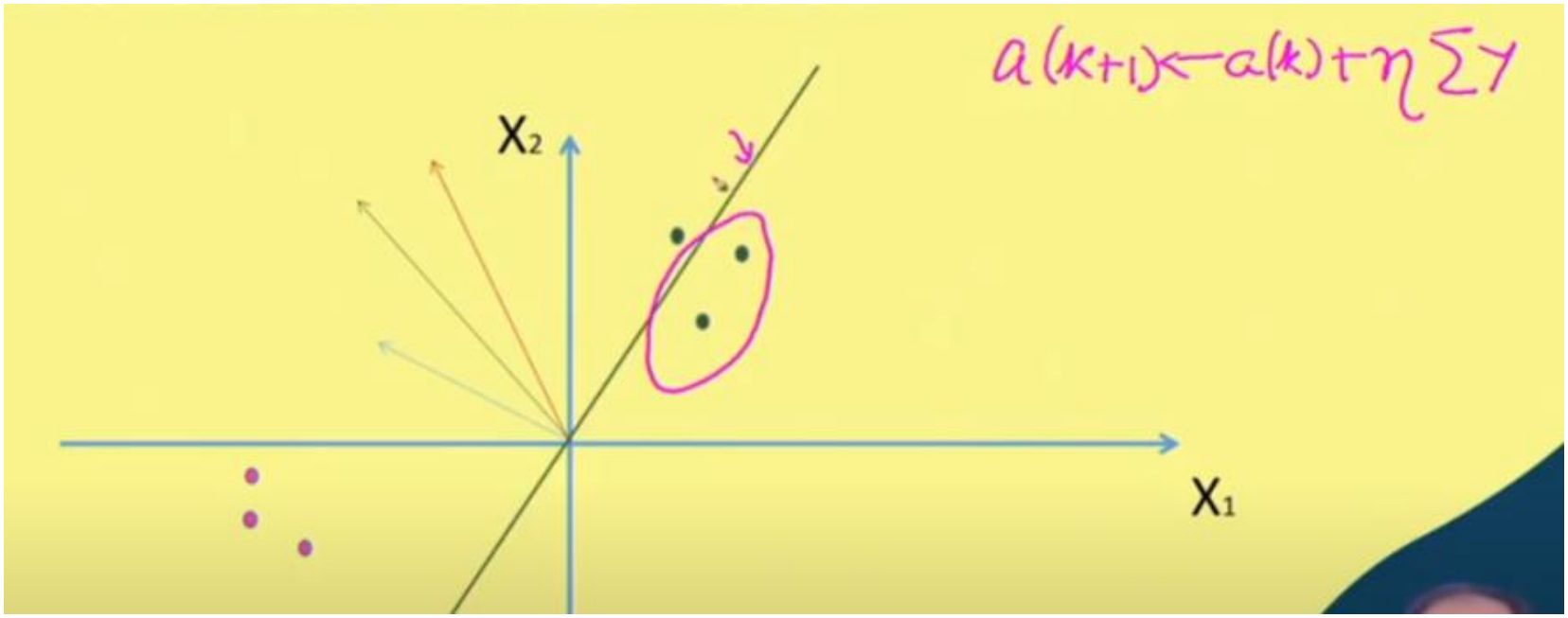
- Suppose, initial weight vector a(0) is chosen something like this



Linear Classifier: Perceptron

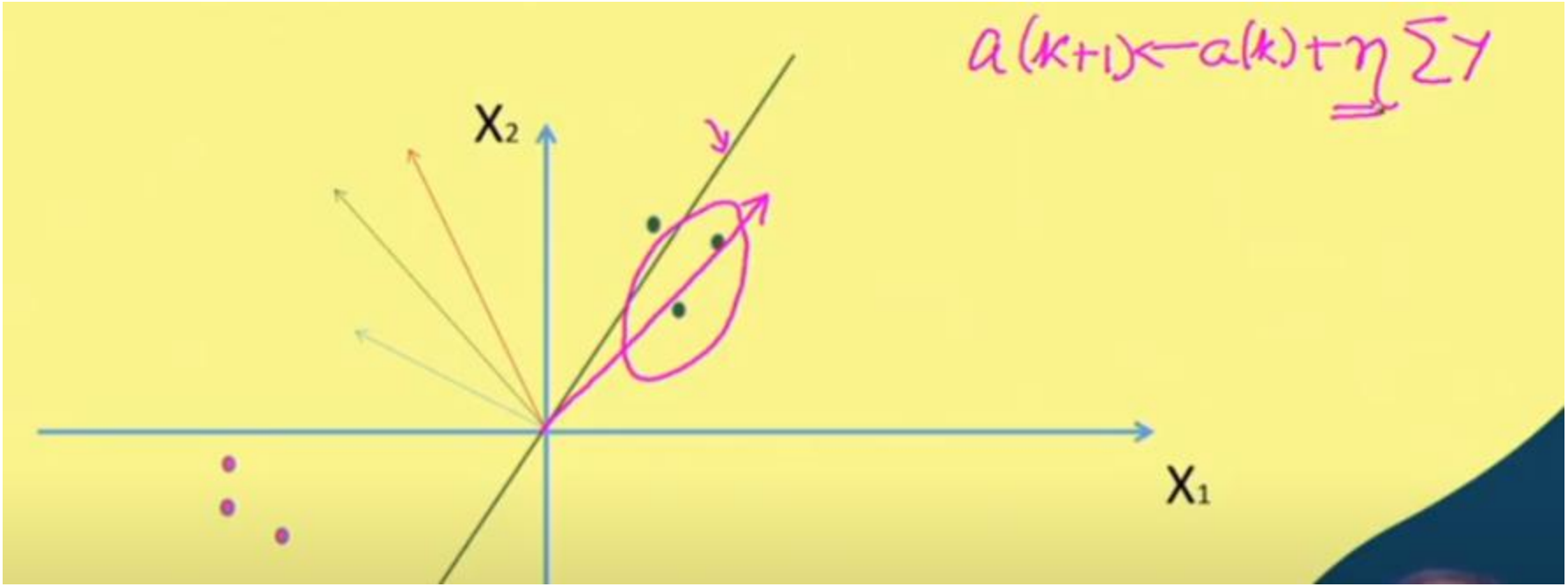


Linear Classifier: Perceptron



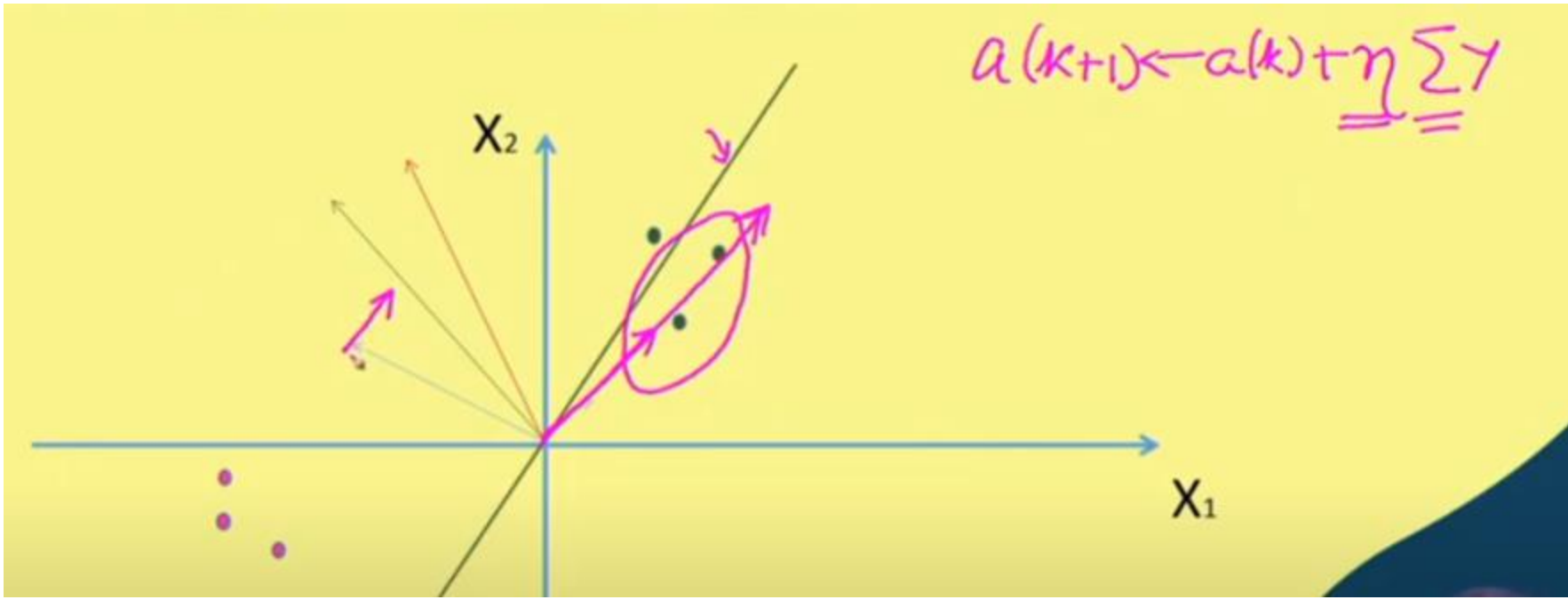
Linear Classifier: Perceptron

- All mis-classified samples are in the direction shown



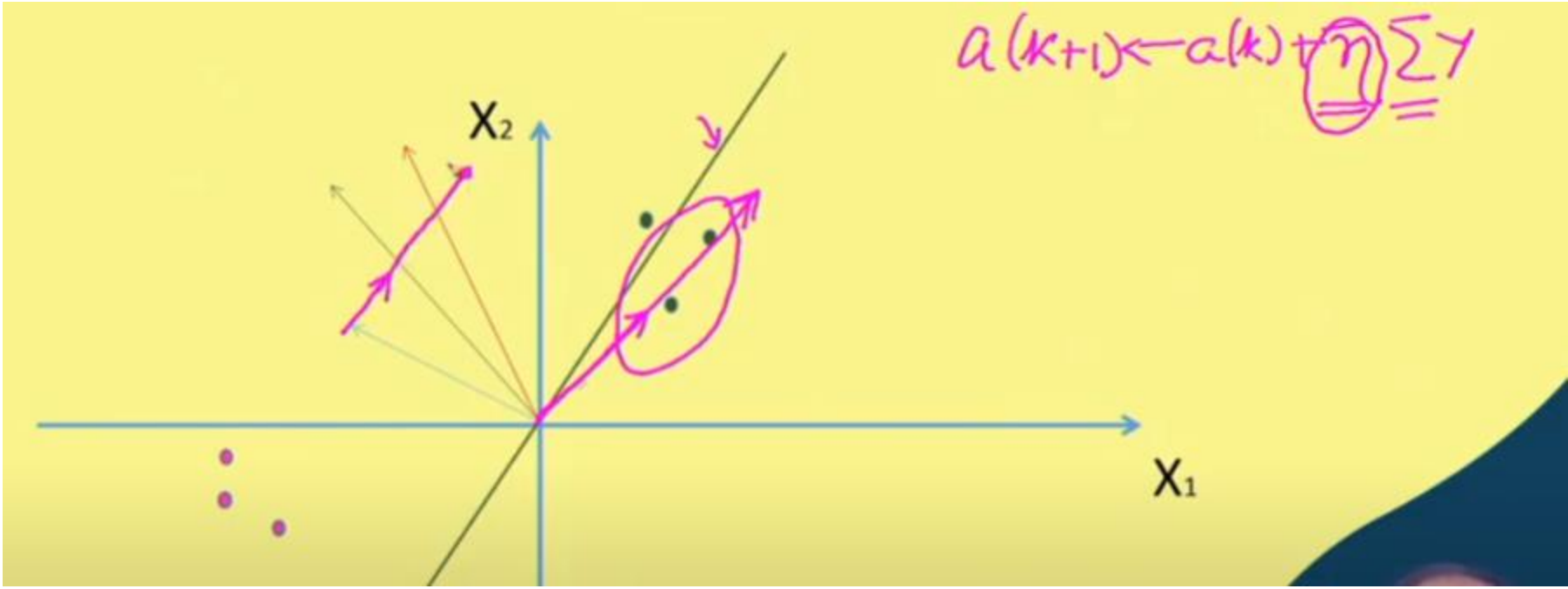
Linear Classifier: Perceptron

- So, a(0) is moved in the direction as shown and slowly it land in the solution region.



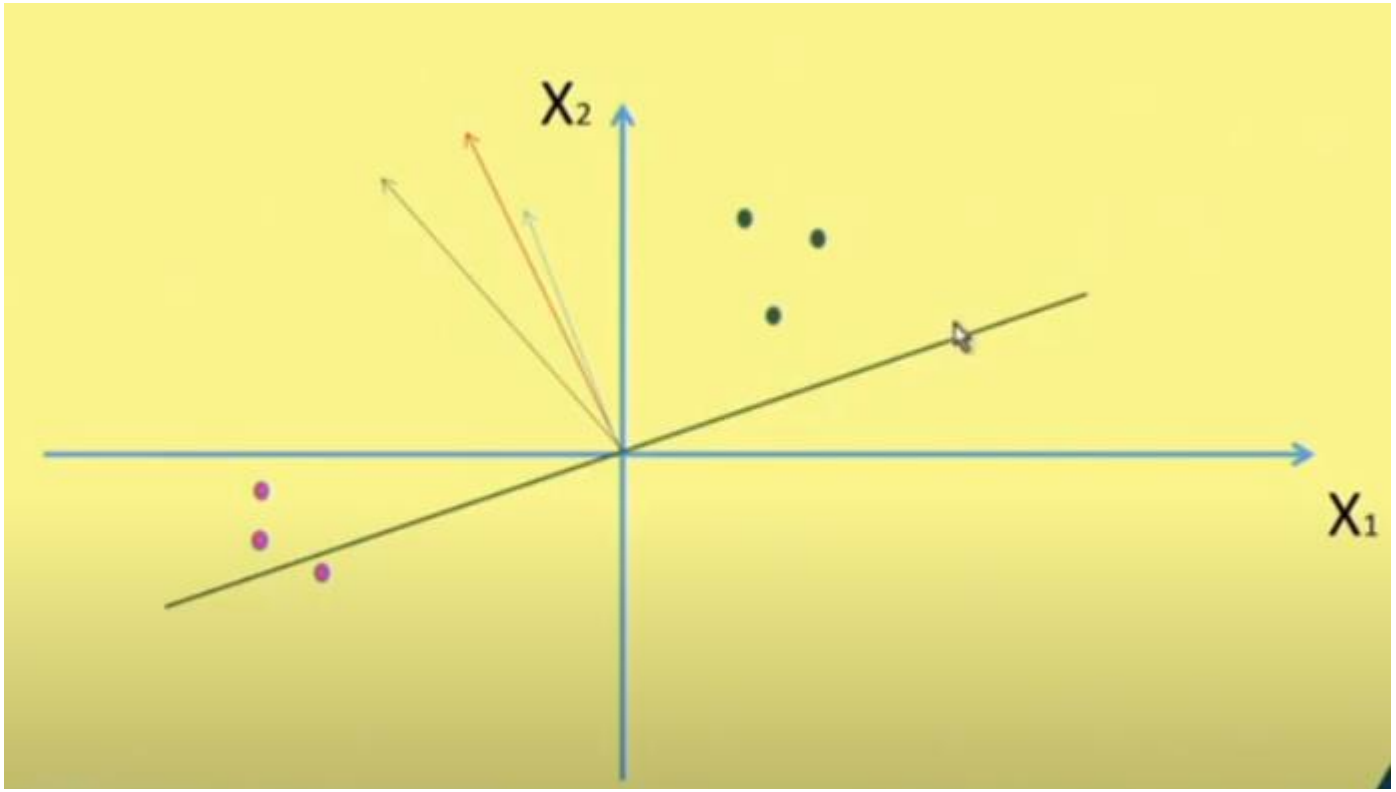
Linear Classifier: Perceptron

- If the learning rate η is very large then solution vector is over shooted.

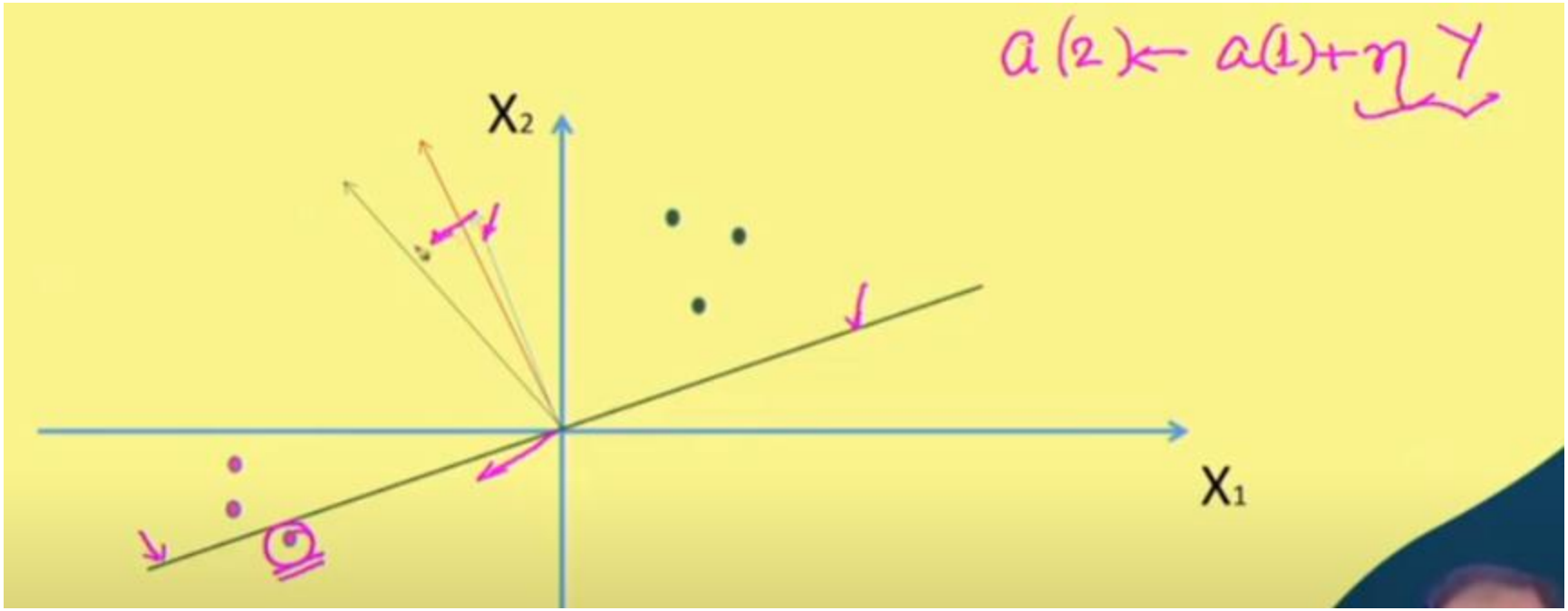


Linear Classifier: Perceptron

- If the learning rate η is very large then solution vector is crossed the solution region.

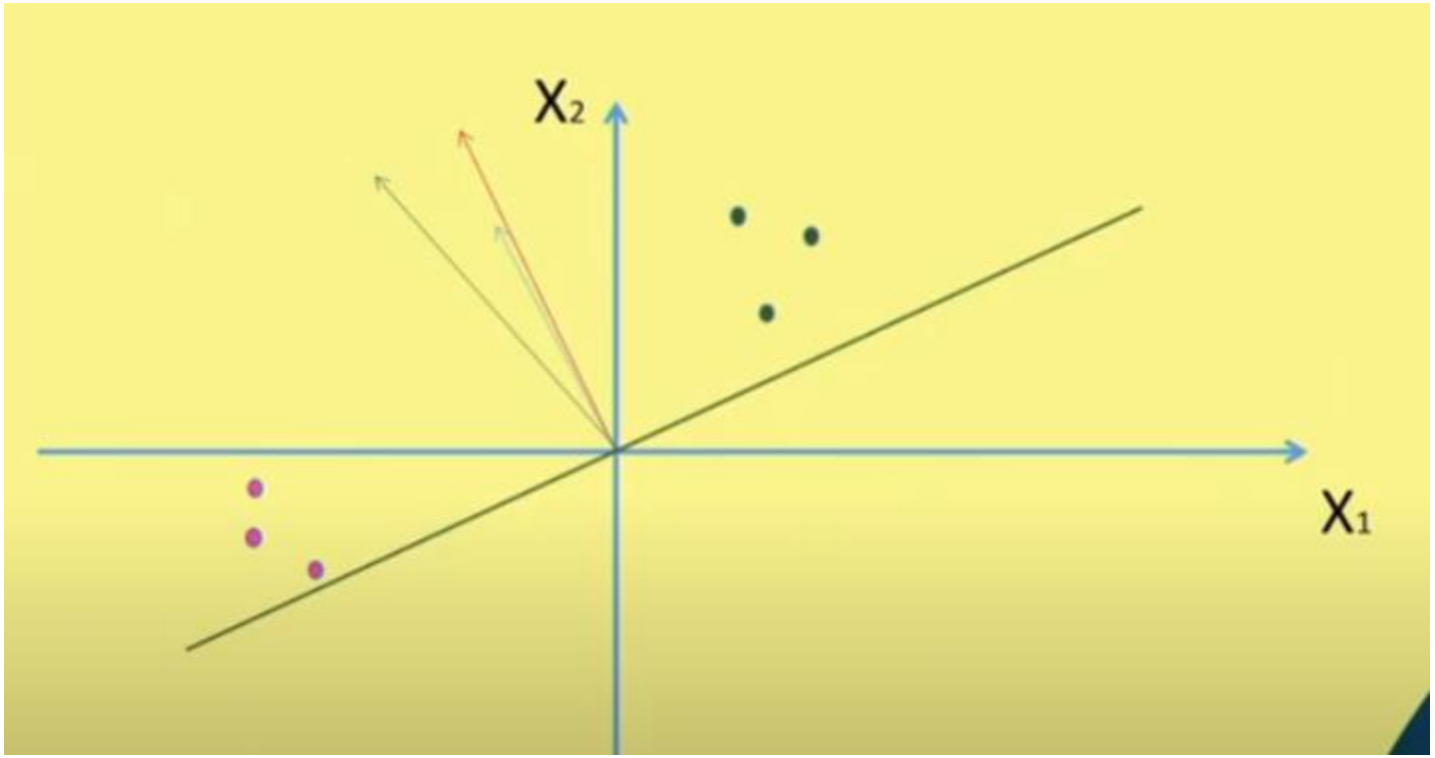


Linear Classifier: Perceptron



Linear Classifier: Perceptron

- This approach clearly says that by taking gradient descent approach it is possible the weight vector converge in the solution region.



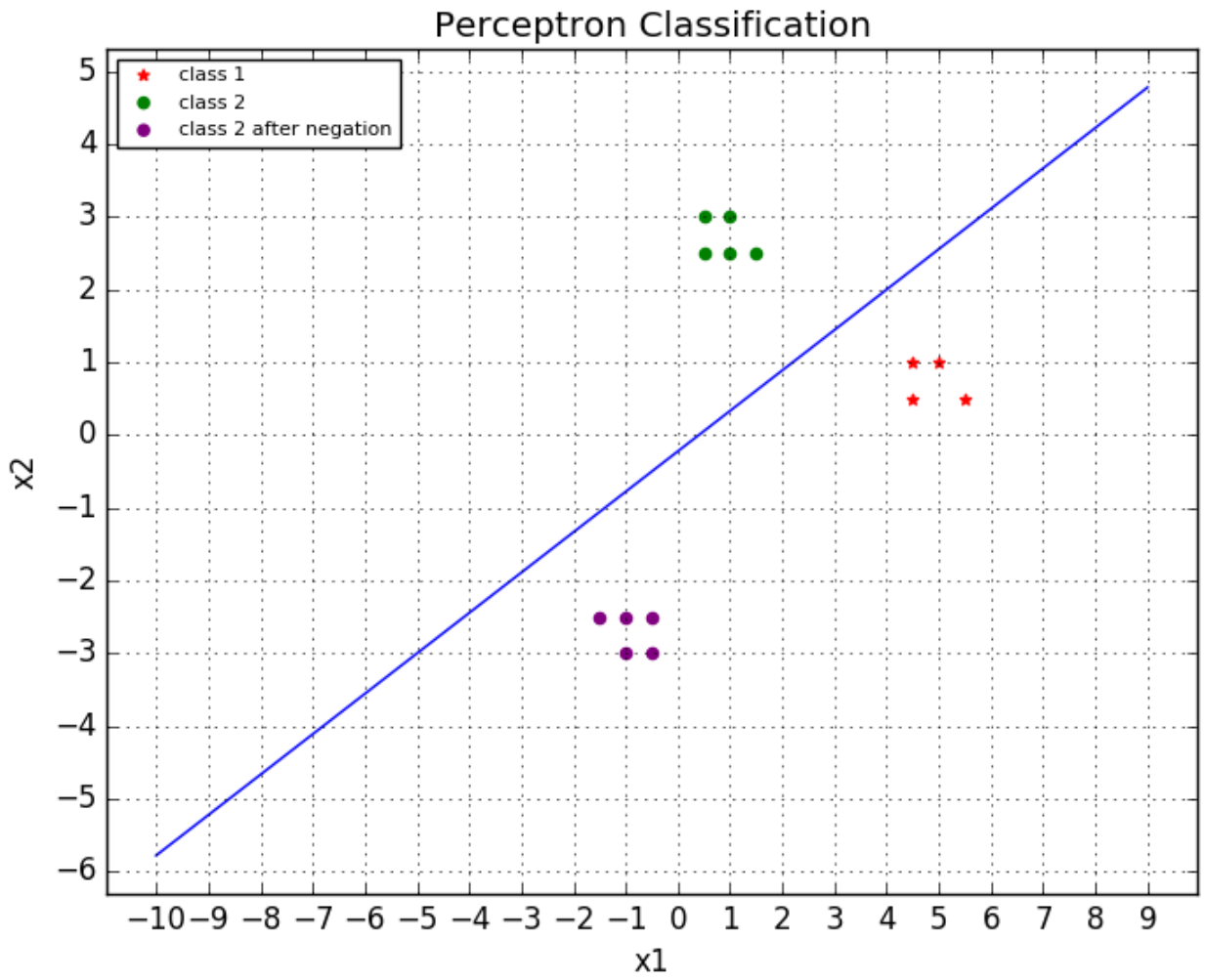
Perceptron criteria

- The approach can be $J(a) = \sum - (a^T y)$
- Using **gradient decent algorithm**
- It works well if the **classes are linearly separable**

MSE criteria

- If the classes are linearly separable, then it is possible to have linear decision boundary like this
- If the classes are not linearly separable then the approach is based on minimum error criteria (Mean Square Error criteria)

Linear Classifier: Perceptron



Summary

- Convergence Procedure of Perceptron Algorithm

Thank you