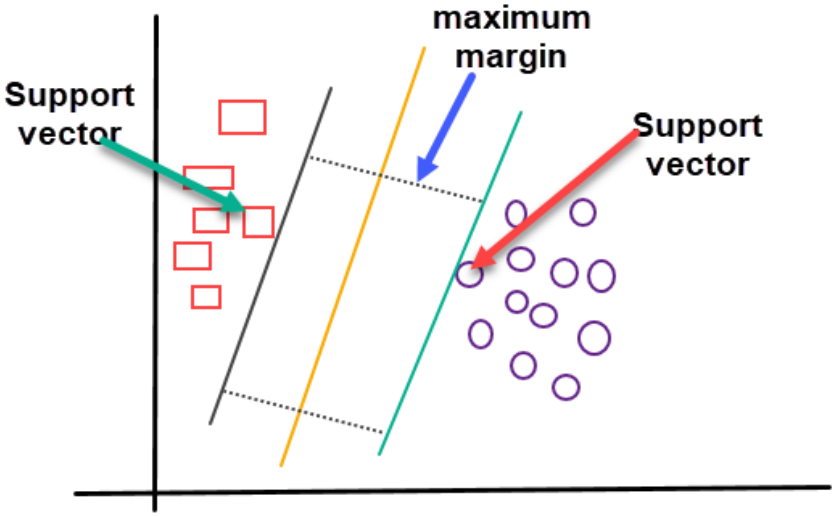
Support Vector Machine

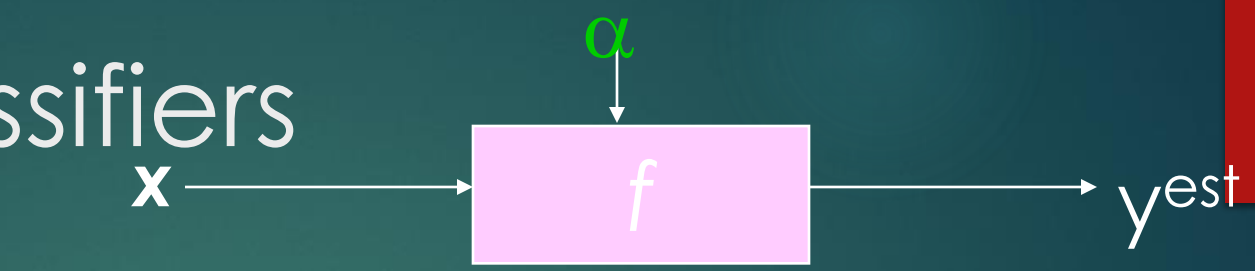
- Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. The main idea behind SVM is to find a boundary (or "hyperplane") that best separates different classes of data points.

- **Key Concepts of SVM**

1. **Hyperplane**: The line (or plane in higher dimensions) that separates different classes.
2. **Support Vectors**: The closest points to the hyperplane from each class. These are the most important points in SVM because they define the position and orientation of the hyperplane.
3. **Margin**: The distance between the hyperplane and the nearest support vector from each class. SVM tries to maximize this margin.



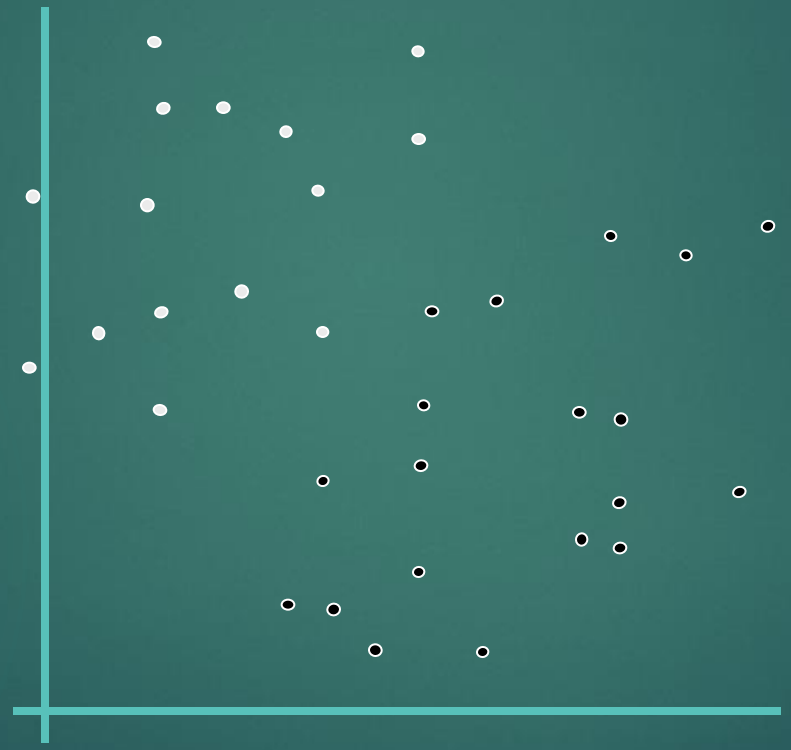
Linear Classifiers



$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

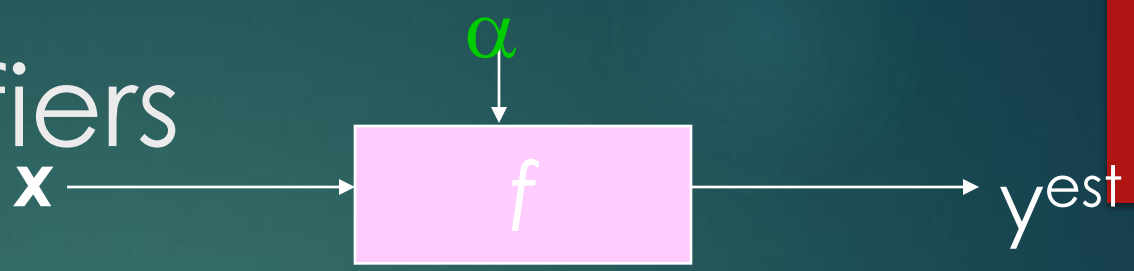
Deep Knowledge

- Class +1
- Class -1

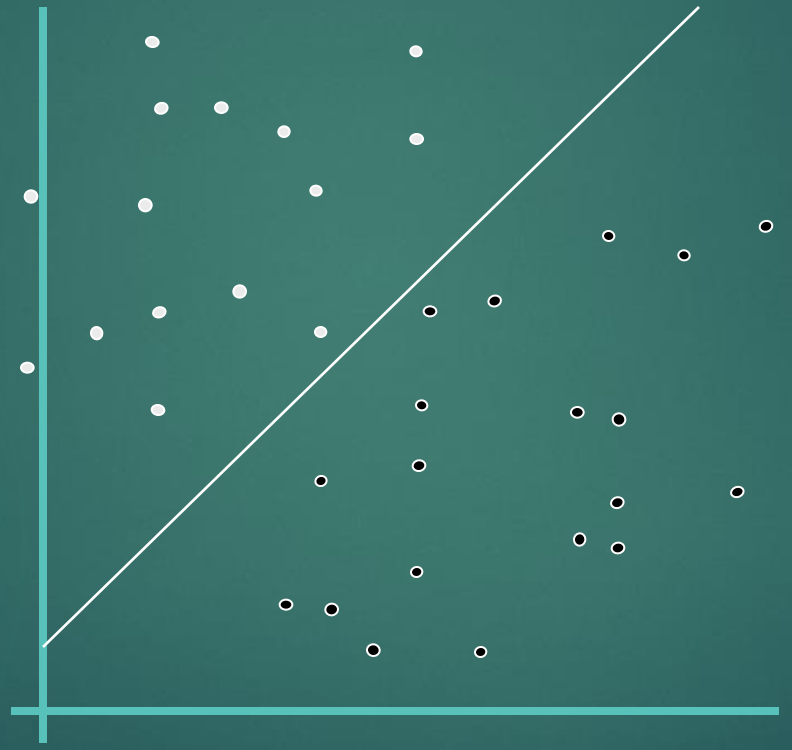


How would you classify this data?

Linear Classifiers

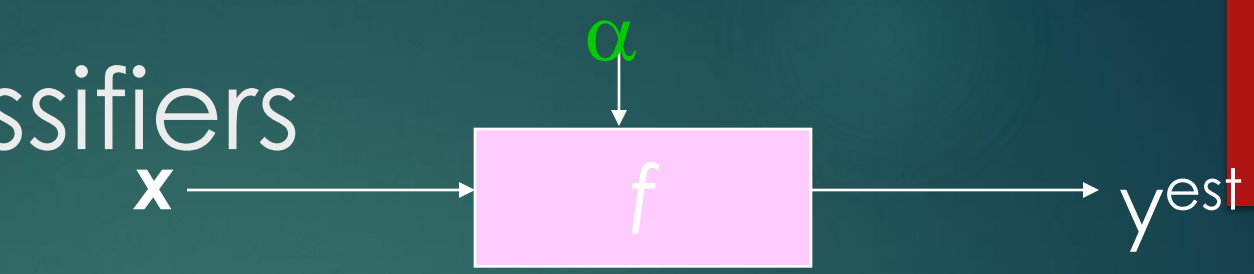


$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$



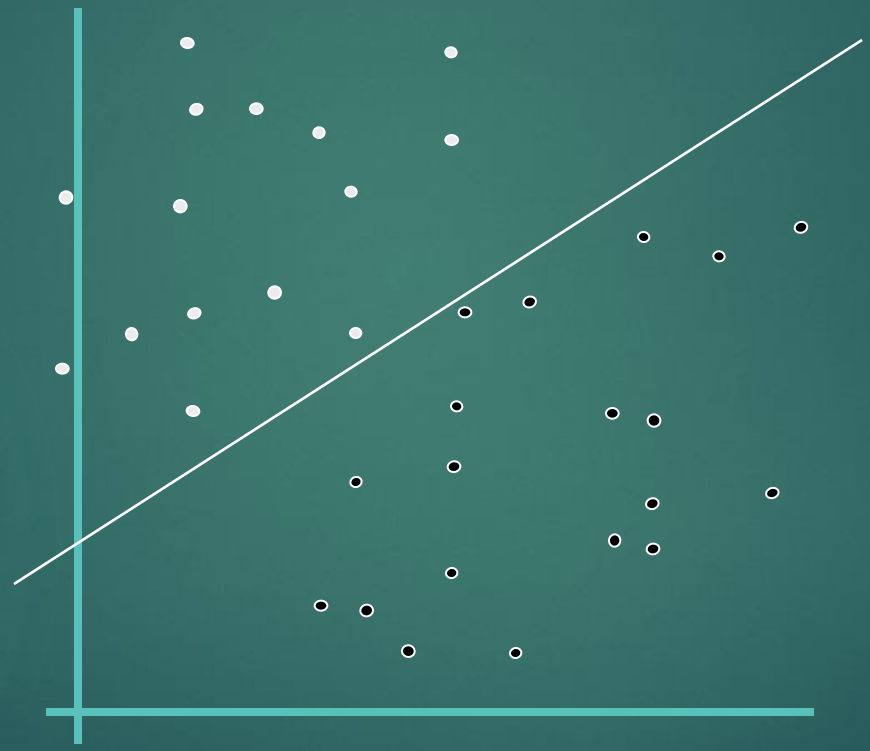
How would you classify this data?

Linear Classifiers



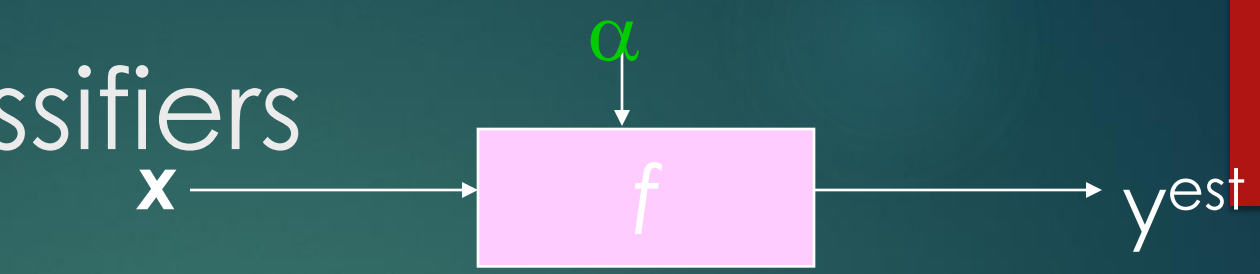
$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- Class +1
- Class -1



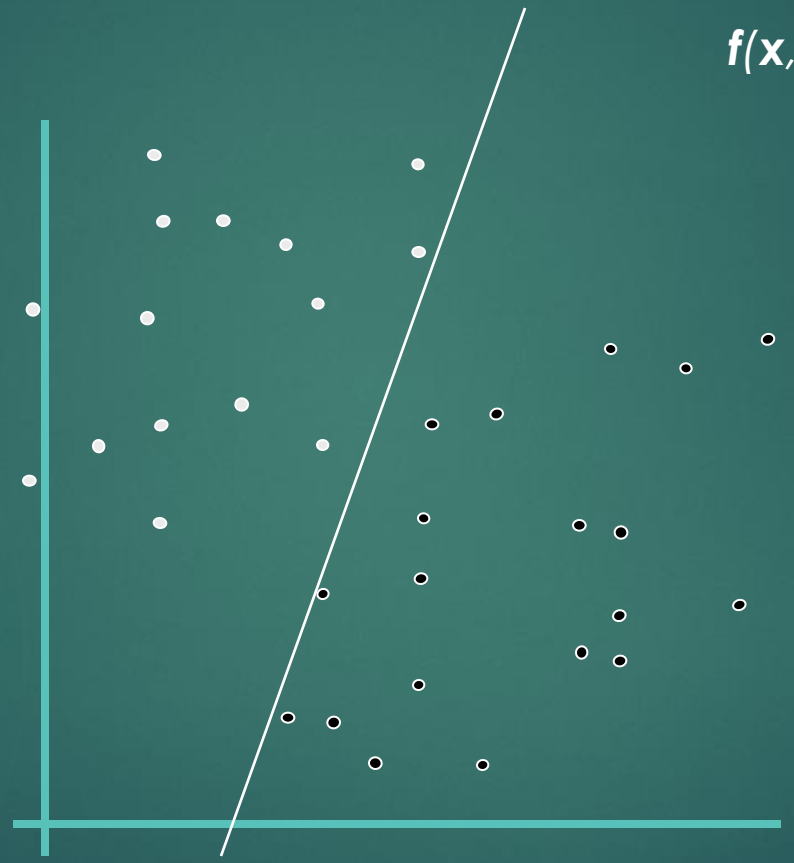
How would you classify this data?

Linear Classifiers



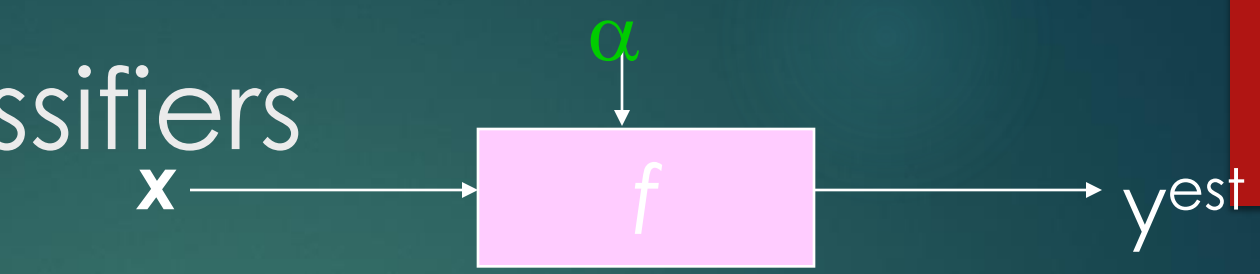
$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- Class +1
- Class -1



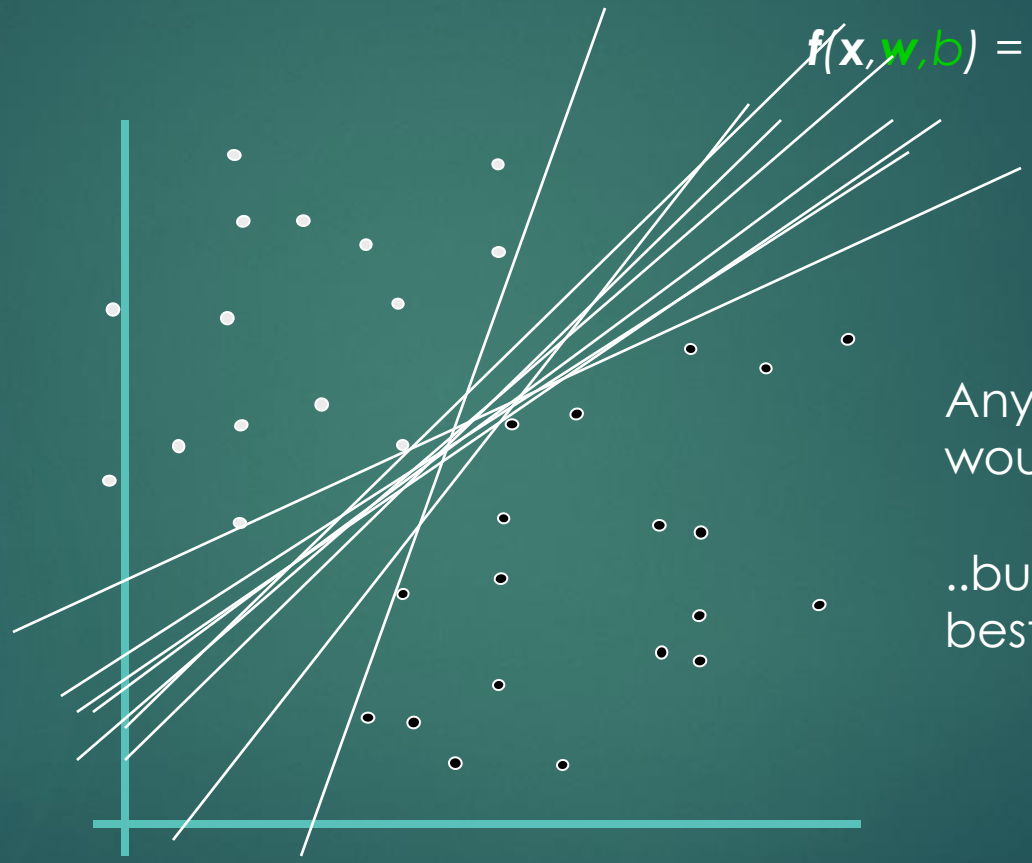
How would you classify this data?

Linear Classifiers



$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

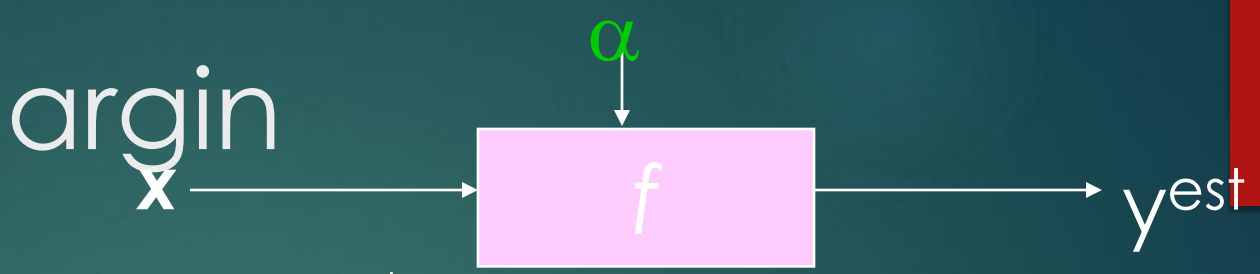
- Class +1
- Class -1



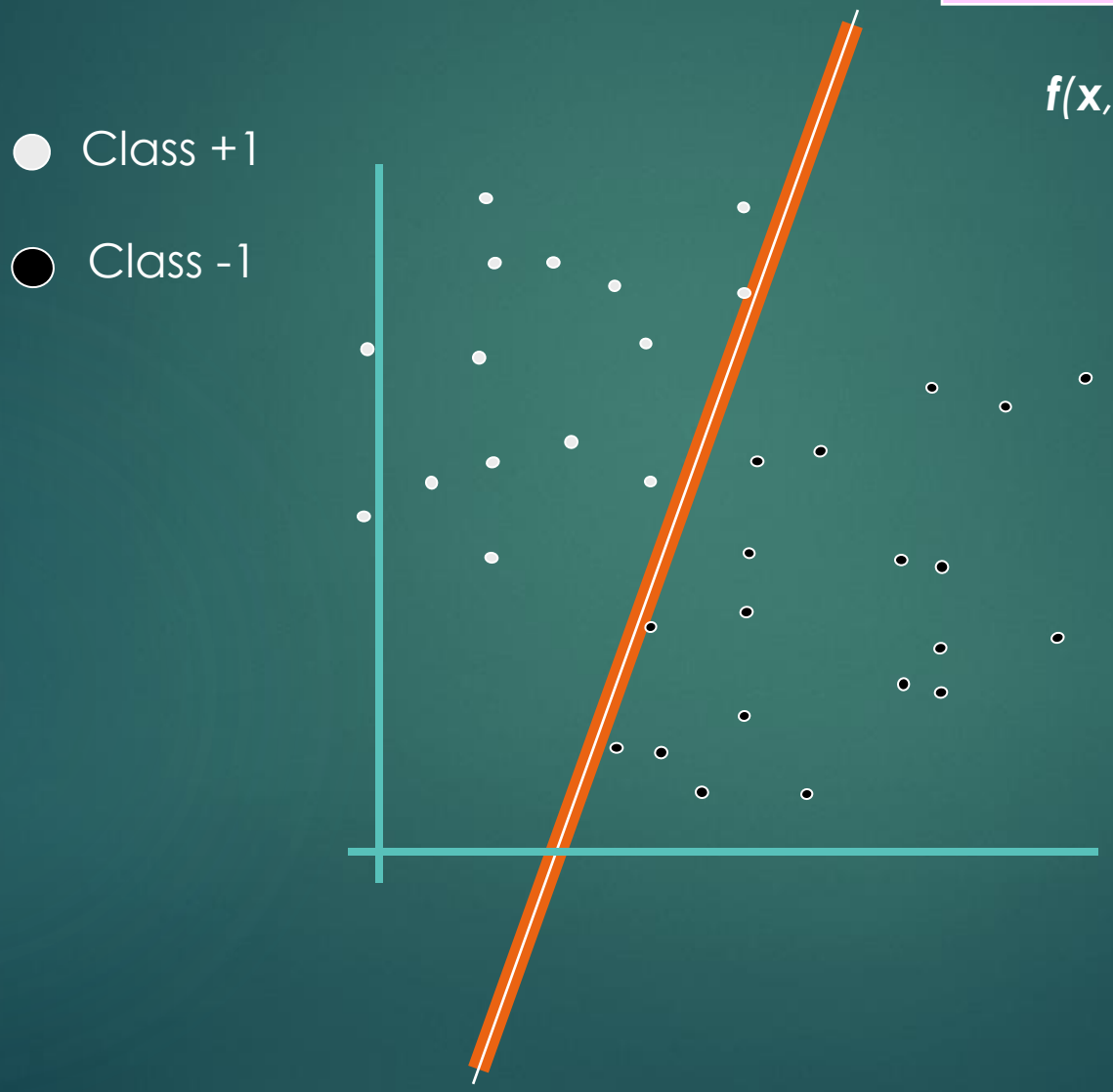
Any of these would be fine..

..but which is best?

Classifier Margin

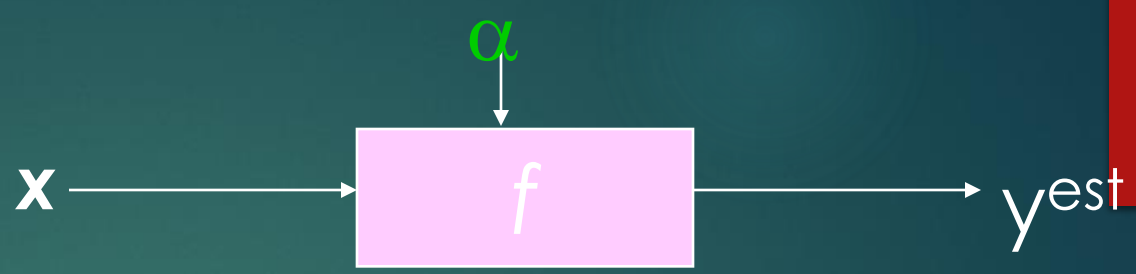


$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$



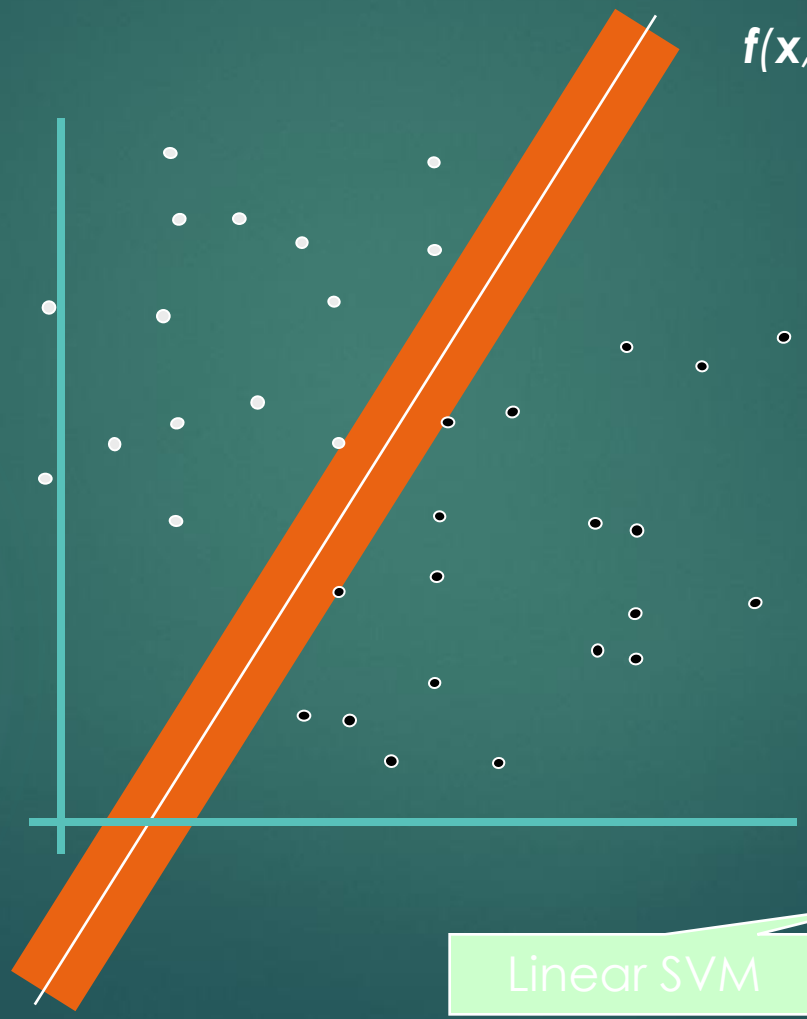
Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



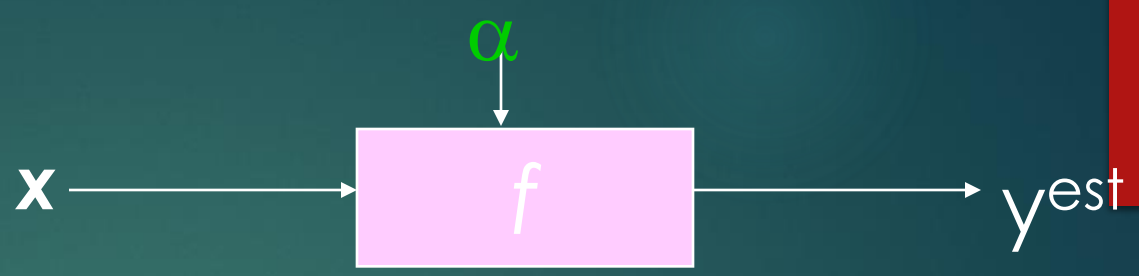
$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- Class +1
- Class -1



The maximum margin linear classifier is the linear classifier with the, um, maximum margin.
This is the simplest kind of SVM (Called an LSVM)

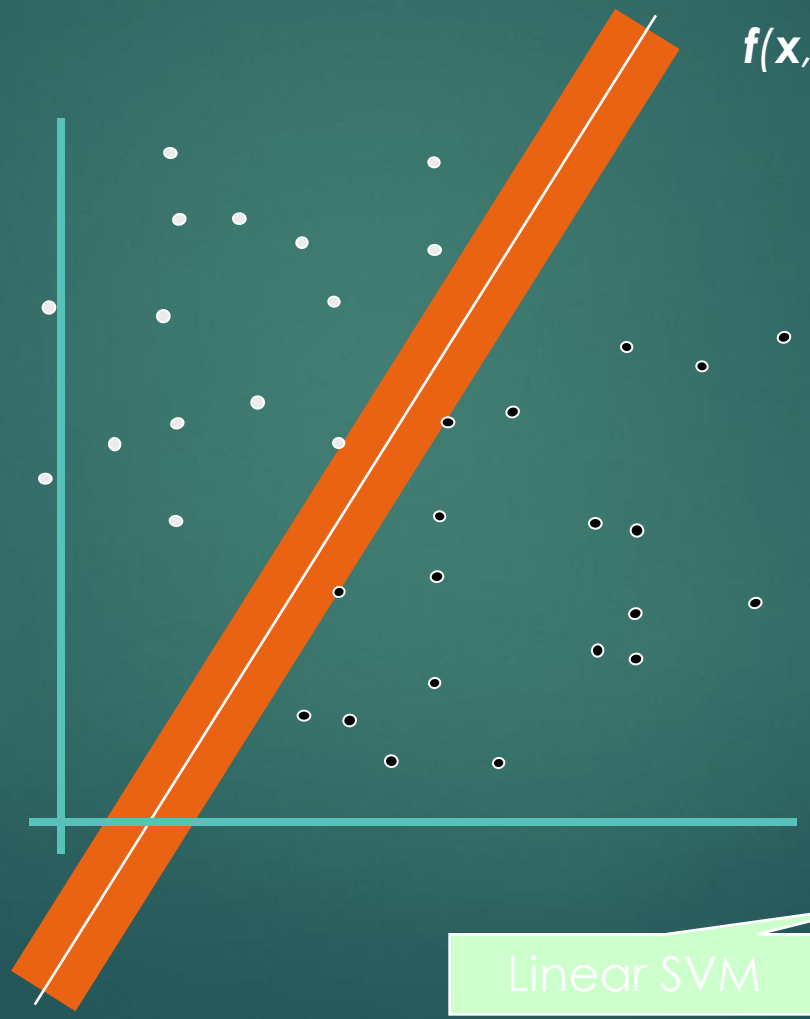
Maximum Margin



$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- Class +1
- Class -1

Support Vectors are those datapoints that the margin pushes up against



The maximum margin linear classifier is the linear classifier with the, um, maximum margin.
This is the simplest kind of SVM (Called an LSVM)

Linear SVM

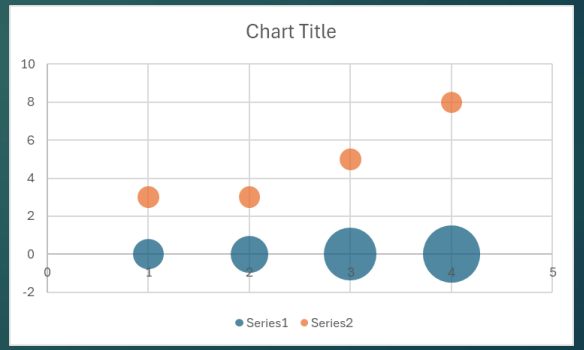
Example

Deep Knowledge

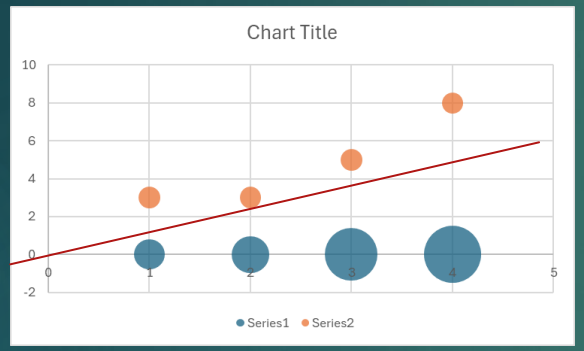
Point (Rows)	X1 (Feature1)	X2 (Feature2)	Class
A	2	3	+1 (Class 1)
B	3	3	+1 (Class 1)
C	6	5	-1 (Class 2)
D	7	8	-1 (Class 2)

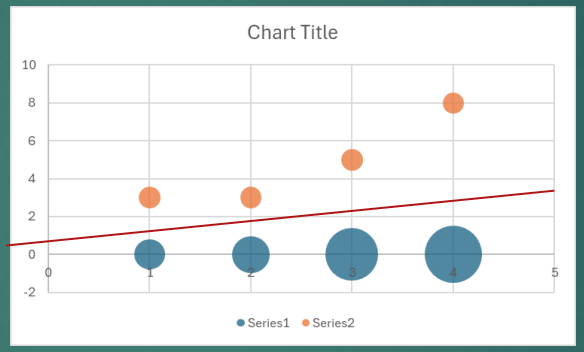
- X-axis: Feature 1 (X1)
- Y-axis: Feature 2 (X2)
- Class 1 (+1) points: A and B

- Class 2 (-1) points: C and D

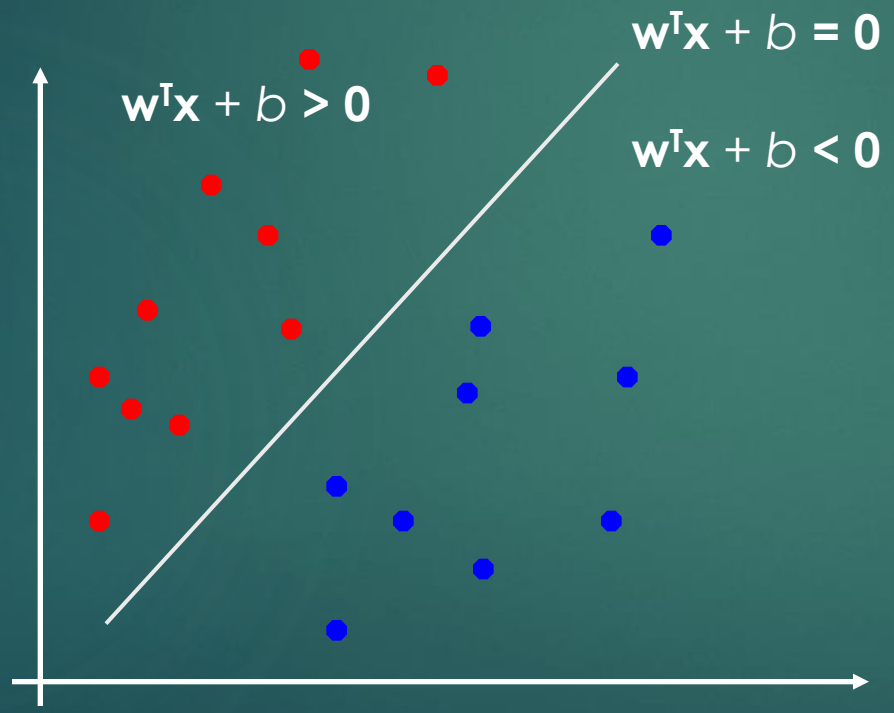


Deep Knowledge





- Binary classification can be viewed as the task of separating classes in feature space:



Dataset

Let's use a small dataset with two classes (`+1` and `-1`) and two features (`X1` and `X2`):

Point	X1	X2	Class
A	2	3	+1
B	4	1	+1
C	1	6	-1
D	2	7	-1

Hyperparameters

- Learning rate (`learning_rate`): 0.01
- Regularization strength (`regularization_strength`): 0.01
- Number of epochs: 2 (we'll demonstrate only one or two)

Initialize Parameters

- **Weights**: Start with weights as `[0, 0]` (for `X1` and `X2`).
- **Bias**: Start with bias as `0`.

Step-by-Step Example (Epochs 1 and 2)

Step 1: Initial Setup

- **Initial Weights**: `[0, 0]`
- **Initial Bias**: `0`

Epoch 1

1. **Point A** (X1 = 2, X2 = 3, Class = +1)
   - **Calculate Decision Function**:

     $$\text{condition} = \text{Class} \times (\text{weights} \cdot \text{features} + \text{bias}) = +1 \times ((0 \times 2) + (0 \times 3) + 0) = 0$$

   - **Update Weights and Bias**:
     - Since `condition < 1`, Point A is misclassified.
     - Apply updates:
       - **Weight Update**:

         $$\text{weights} = \text{weights} - \text{learning rate} \times (2 \times \text{regularization strength} \times \text{weights} - \text{Class} \times \text{features})$$

         Substituting values:

         $$\text{weights} = [0, 0] - 0.01 \times (2 \times 0.01 \times [0, 0] - (+1) \times [2, 3]) = [0.02, 0.03]$$

       - **Bias Update**:

         $$\text{bias} = \text{bias} - \text{learning rate} \times (-\text{Class})$$

         Substituting values:

         $$\text{bias} = 0 - 0.01 \times (-1) = 0.01$$

   - **Updated Parameters**:
     - Weights: `[0.02, 0.03]`
     - Bias: `0.01`

Dataset

Let's use a small dataset with two classes (`+1` and `-1`) and two features (`X1` and `X2`):

Point	X1	X2	Class
A	2	3	+1
B	4	1	+1
C	1	6	-1
D	2	7	-1

Hyperparameters

- Learning rate (`learning_rate`): 0.01
- Regularization strength (`regularization_strength`): 0.01
- Number of epochs: 2 (we'll demonstrate only one or two)

Initialize Parameters

- **Weights**: Start with weights as `[0, 0]` (for `X1` and `X2`).
- **Bias**: Start with bias as `0`.

Step-by-Step Example (Epochs 1 and 2)

Step 1: Initial Setup

- **Updated Parameters:**
  - Weights: `[0.02, 0.03]`
  - Bias: `0.01`

2. **Point B** (X1 = 4, X2 = 1, Class = +1)

- **Calculate Decision Function:**

$$\text{condition} = +1 \times ((0.02 \times 4) + (0.03 \times 1) + 0.01) = 0.12$$

- **Update Weights and Bias:**
  - Since `condition < 1`, Point B is misclassified.
  - Apply updates:
    - **Weight Update:**

$$\text{weights} = [0.02, 0.03] - 0.01 \times (2 \times 0.01 \times [0.02, 0.03] - (+1) \times [4, 1]) = [0.06, 0.04]$$

    - **Bias Update:**

$$\text{bias} = 0.01 - 0.01 \times (-1) = 0.02$$

- **Updated Parameters:**
  - Weights: `[0.06, 0.04]`
  - Bias: `0.02`

Deep Knowledge

Dataset

Let's use a small dataset with two classes (`+1` and `-1`) and two features (`X1` and `X2`):

Point	X1	X2	Class
A	2	3	+1
B	4	1	+1
C	1	6	-1
D	2	7	-1

Hyperparameters

- Learning rate (`learning_rate`): 0.01
- Regularization strength (`regularization_strength`): 0.01
- Number of epochs: 2 (we'll demonstrate only one or two)

Initialize Parameters

- **Weights**: Start with weights as `[0, 0]` (for `X1` and `X2`).
- **Bias**: Start with bias as `0`.

Step-by-Step Example (Epochs 1 and 2)

- **Updated Parameters:**
  - Weights: `[0.06, 0.04]`
  - Bias: `0.02`

3. **Point C** (X1 = 1, X2 = 6, Class = -1)

- **Calculate Decision Function:**

$$\text{condition} = -1 \times ((0.06 \times 1) + (0.04 \times 6) + 0.02) = -0.34$$

- **Update Weights and Bias:**
  - Since `condition < 1`, Point C is misclassified.
  - Apply updates:
    - **Weight Update:**

$$\text{weights} = [0.06, 0.04] - 0.01 \times (2 \times 0.01 \times [0.06, 0.04] - (-1) \times [1, 6]) = [0.07, 0.10]$$

    - **Bias Update:**

$$\text{bias} = 0.02 - 0.01 \times (+1) = 0.01$$

- **Updated Parameters:**
  - Weights: `[0.07, 0.10]`
  - Bias: `0.01`

3. **Point C** (X1 = 1, X2 = 6, Class = -1)
   - **Calculate Decision Function**:
     $$\text{condition} = -1 \times ((0.06 \times 1) + (0.04 \times 6) + 0.02) = -0.34$$
   - **Update Weights and Bias**:
     - Since `condition < 1`, Point C is misclassified.
     - Apply updates:
       - **Weight Update**:
         $$\text{weights} = [0.06, 0.04] - 0.01 \times (2 \times 0.01 \times [0.06, 0.04] - (-1) \times [1, 6]) = [0.07, 0.10]$$
       - **Bias Update**:
         $$\text{bias} = 0.02 - 0.01 \times (+1) = 0.01$$
   - **Updated Parameters**:
     - Weights: `[0.07, 0.10]`
     - Bias: `0.01`
4. **Point D** (X1 = 2, X2 = 7, Class = -1)
   - **Calculate Decision Function**:
     $$\text{condition} = -1 \times ((0.07 \times 2) + (0.10 \times 7) + 0.01) = -0.78$$
   - **Update Weights and Bias**:
     - Since `condition < 1`, Point D is misclassified.
     - Apply updates:
       - **Weight Update**:
         $$\text{weights} = [0.07, 0.10] - 0.01 \times (2 \times 0.01 \times [0.07, 0.10] - (-1) \times [2, 7]) = [0.09, 0.17]$$
       - **Bias Update**:
         $$\text{bias} = 0.01 - 0.01 \times (+1) = 0.0$$
   - **Updated Parameters**:
     - Weights: `[0.09, 0.17]`
     - Bias: `0.0`

Epoch 2

Repeat the same steps for each point again, using the updated weights and bias from Epoch 1.

Here's a summary:

1. **Point A**:
   - Condition: $+1 \times ((0.09 \times 2) + (0.17 \times 3) + 0) = 0.77$ (misclassified).
   - Updated Weights: `[0.11, 0.20]`
   - Updated Bias: `0.01`
2. **Point B**:
   - Condition: $+1 \times ((0.11 \times 4) + (0.20 \times 1) + 0.01) = 0.66$ (misclassified).
   - Updated Weights: `[0.15, 0.21]`
   - Updated Bias: `0.02`
3. **Point C**:
   - Condition: $-1 \times ((0.15 \times 1) + (0.21 \times 6) + 0.02) = -1.33$ (correctly classified).
   - No update needed for weights or bias.
4. **Point D**:
   - Condition: $-1 \times ((0.15 \times 2) + (0.21 \times 7) + 0.02) = -1.79$ (correctly classified).
   - No update needed for weights or bias.

Final Parameters After Epoch 2

- **Weights**: `[0.15, 0.21]`
- **Bias**: `0.02`