# Assignment#4: Perceptron

- Part 1
  - Plot data from data.csv
  - Implement perceptron using the heuristic approach (left box next page)
  - Plot the initial separation line as red, subsequent ones after each iteration in dashed green, and the last one in black (see page 4)
  - Play with the learning rate
  - Analysis the results in a report

- Part 2
  - Plot data from data.csv
  - Implement perceptron using the Gradient Descent approach (right box next page)
  - Play with learning rate, number of epochs.
  - Plot the initial separation line as red, subsequent ones after each iteration in dashed green, and the last one in black
  - Compute log loss (error) and plot the error graph every 10 epoch (see page 5)
  - Analysis the results in a report

# Learning by Gradient Descent (Right Side)

1. Start a perceptron with random weights and bias: $w_1, w_2, \ldots, w_n, b$
2. For **each of** all points (data) with their corresponding labels (answers):
   - 2.1. Classify according to the perceptron
   - 2.2. For a misclassified point $(x_1, x_2, \ldots, x_n)$ :
     - 2.2.1. If classification==0:
       - 2.2.1.1. $b + r \rightarrow b$
       - 2.2.1.2. For all $w_i$: $w_i + rx_i \rightarrow w_i$
     - 2.2.2. If classification==1:
       - 2.2.2.1. $b - r \rightarrow b$
       - 2.2.2.2. For all $w_i$: $w_i - rx_i \rightarrow w_i$
3. Repeat #2 enough number of times

***Earlier heuristic approach with binary classification***

1. Start a perceptron with random weights and bias: $w_1, w_2, \ldots, w_n, b$
2. For **each of** all points (data) with their corresponding labels (answers):
   - 2.1. Compute prediction output ($\hat{y}$)
   - 2.2. Compute error function ($y - \hat{y}$)
   - 2.3. $b + r\mathbf{(y - \hat{y})} \rightarrow b$
   - 2.4. For all $w_i$: $w_i + r\mathbf{(y - \hat{y})}x_i \rightarrow w_i$
3. Repeat #2 until **error is small**

***Note: $\hat{y}$ is no longer 0 or 1 from a step function.***

***See next page***

# Perceptron

- Linear Function → Biased Sum of Weighted input

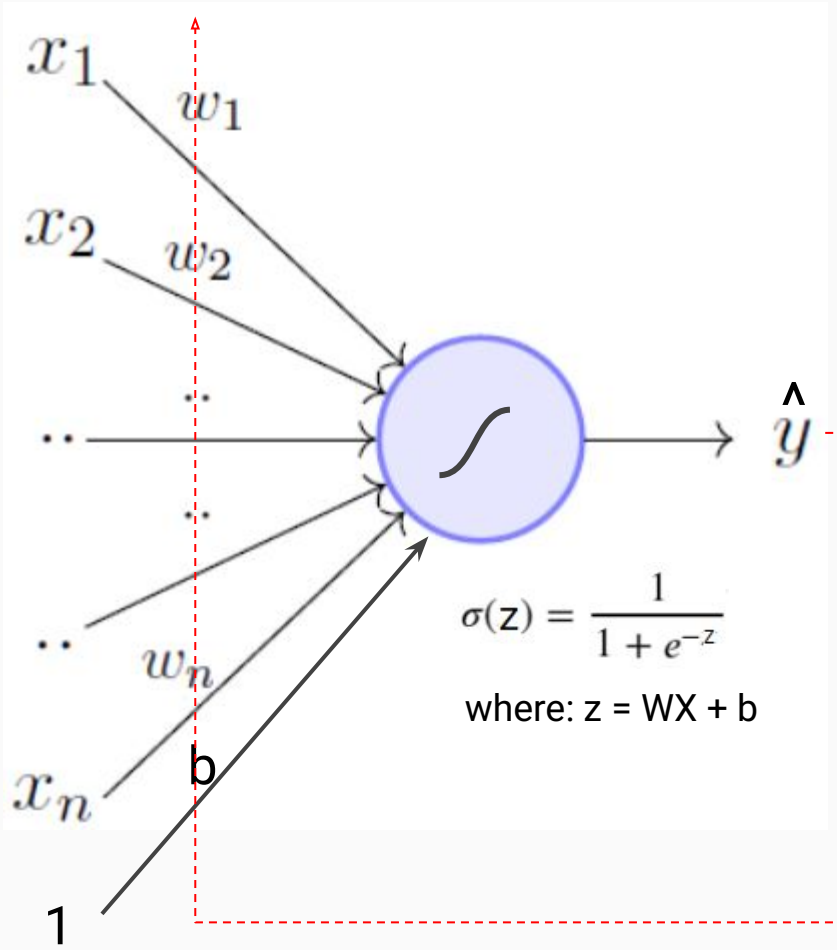
z = WX + b

$$= W \cdot X^t + b$$

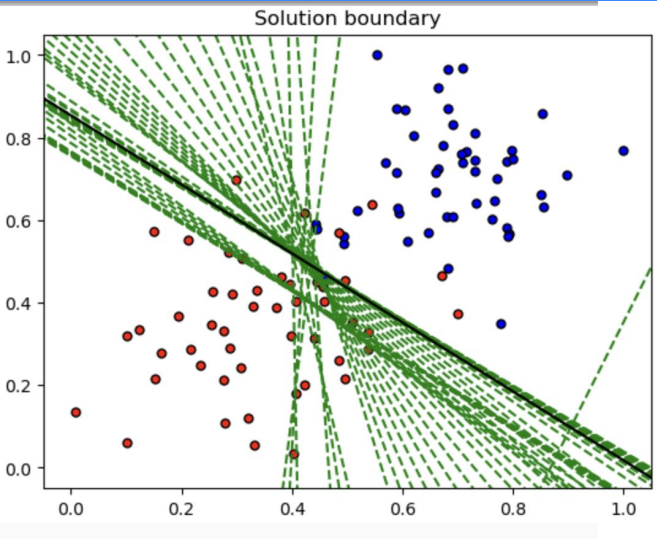
$$= \sum_{i=1}^{n} W_i x_i + b$$

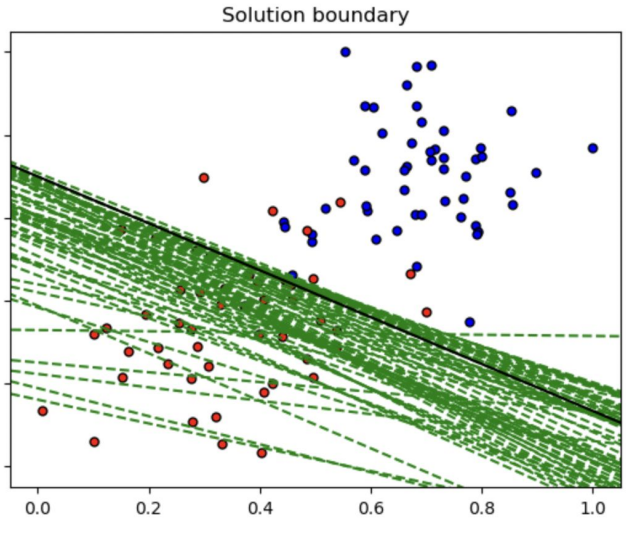
- Sigmoid → **Continuous** Classification

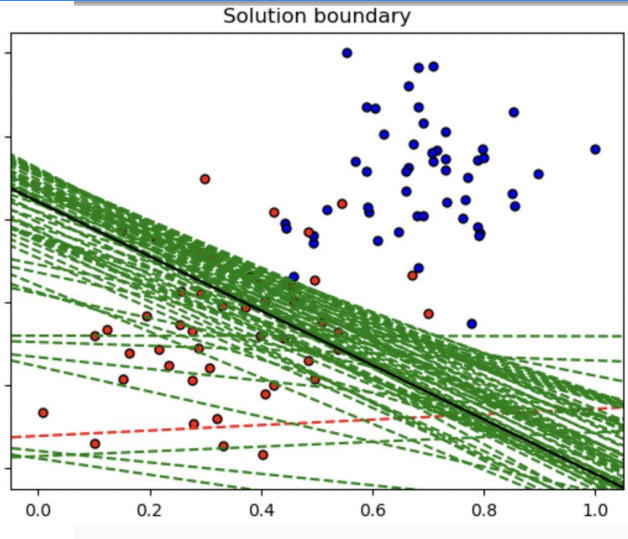
$\hat{y} = \sigma(WX + b)$



# Learning rate (samples, your results may be different)




learning_rate = 0.01

iteration = 65

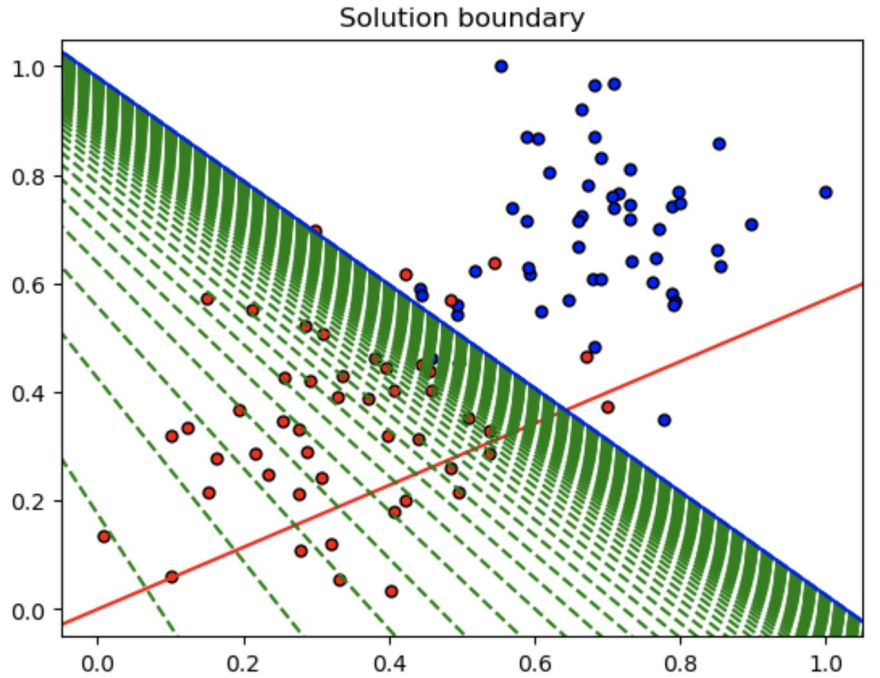
learning_rate = 0.1

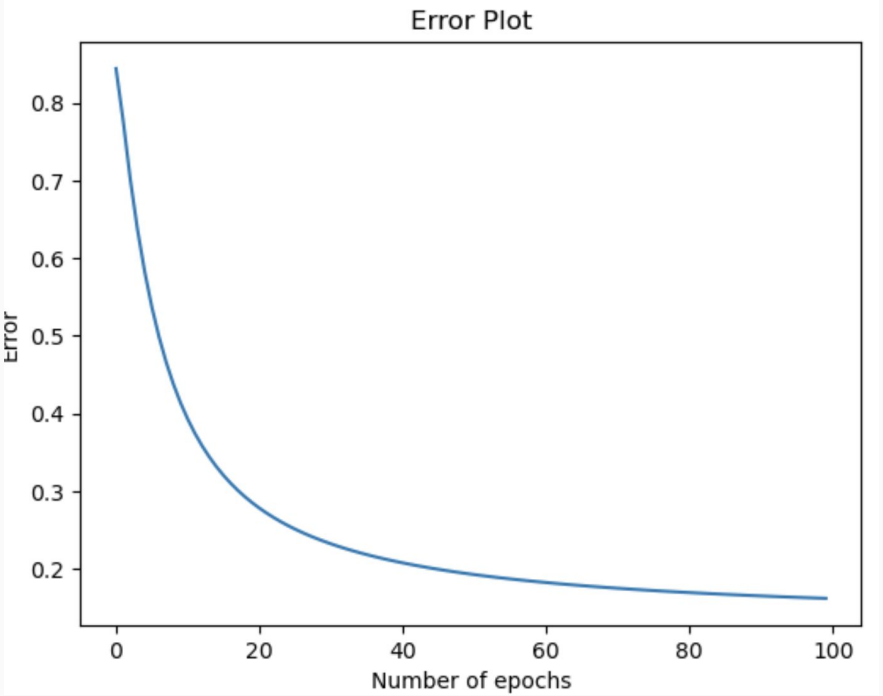
iteration = 65

learning_rate = 1

iteration = 65

# For Part 2 (samples, your results may be different)



# Submission

- In Canvas Assignments:
  - **Submit as File Uploads**: A PDF report with snippets of your code, all the plots and the analysis of them.
  - **Submit as Web URL**: Link to your Github repo/project/ with folder name Assignment_4, which include all your Jupyter Notebook code and any other associated files.
- Make sure you have invited me to your Github repo/project/folder

Grading Rubric (Total 100% of 10 pts in final grade)

- Data Loading (10%)
  - Dataset is correctly retrieved and plotted.
- Part1 Implementation (40%)
  - Experiments with learning rate and number of epochs
  - Analysis of results
- Part 2 Implementation (40%)
  - Experiments with learning rate and number of epochs
  - Analysis of results
- PDF Report (10%)