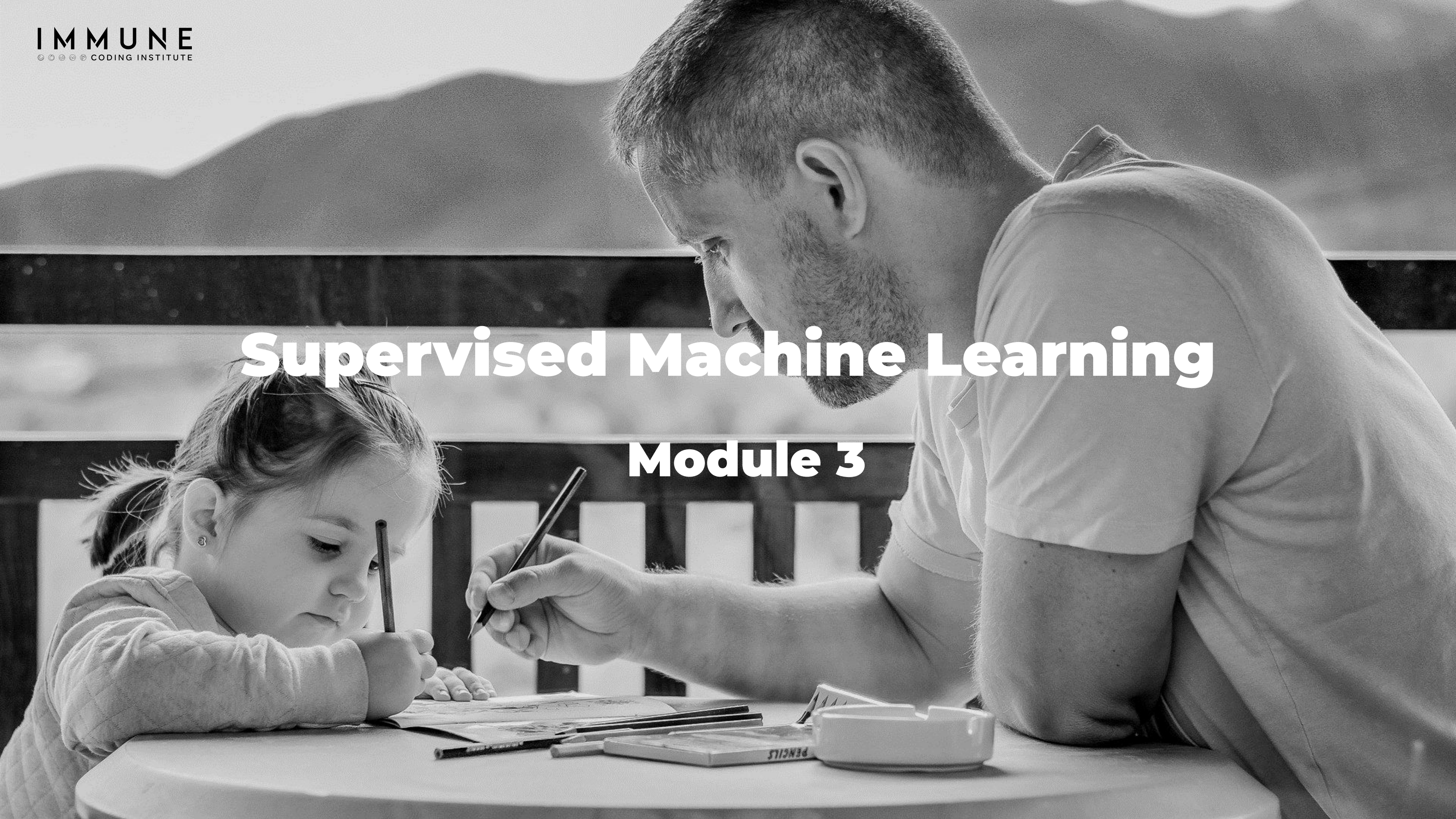


Supervised Machine Learning

Module 3

IMMUNE
CODING INSTITUTE

Module 3 Summary

SESSION	TITLE	TEACHER
1	ML Foundations	Juan
2	Regression Introduction and Practice	Juan
3	**Classification Introduction and Practice**	**Carlos**
4	Feature Engineering and Selection for ML	Carlos
5	Advanced Supervised Models 1	Carlos
6	Advanced Supervised Models 2	Carlos
7	Hands-on Practice	Carlos

IMMUNE CODING INSTITUTE

Introduction to Classification

Outline

- Introduction to Classification
- Definition and Examples
- Logistic Regression
- ROC Curve
- Linear Separability
- Multiclass Classification
- Imbalanced data
- The Bayes Classifier (Naïve Bayes)
- KNN

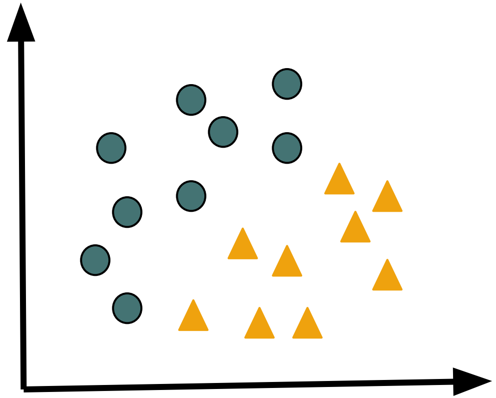
Classification examples

- Email Spam Detection [1]
  https://www.researchgate.net/publication/333677700_Machine_learning_for_email_spam_filtering_review_approaches_and_open_research_problems

- Fraud detection [2]
  https://www.kaggle.com/kabure/credit-card-fraud-prediction-rf-smote

- Predicting bank credit worthiness [3]
  https://www.researchgate.net/publication/326552481_Predictive_Modelling_for_Credit_Risk_Detection_using_Ensemble_Method

- Flower Classifier: more than 2 classes (Multiclass Problem) [4]
  https://www.geeksforgeeks.org/multiclass-classification-using-scikit-learn/



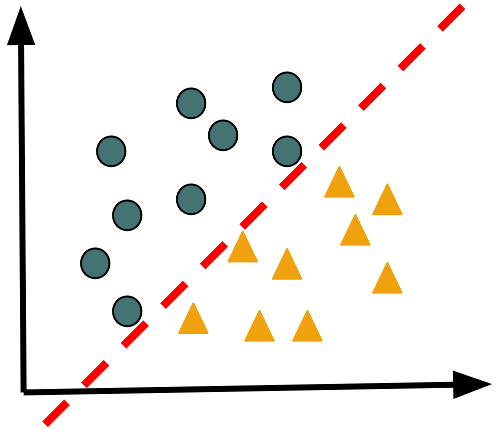
Classification examples

- Email Spam Detection [1]
  https://www.researchgate.net/publication/333677700_Machine_learning_for_email_spam_filtering_review_approaches_and_open_research_problems

- Fraud detection [2]
  https://www.kaggle.com/kabure/credit-card-fraud-prediction-rf-smote

- Predicting bank credit worthiness [3]
  https://www.researchgate.net/publication/326552481_Predictive_Modelling_for_Credit_Risk_Detection_using_Ensemble_Method

- Flower Classifier: more than 2 classes (Multiclass Problem) [4]
  https://www.geeksforgeeks.org/multiclass-classification-using-scikit-learn/



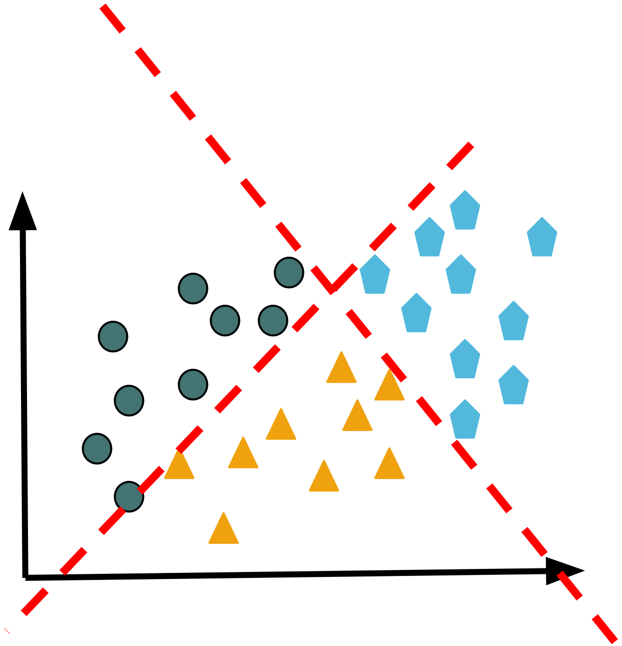
Classification examples

- Email Spam Detection [1]
  https://www.researchgate.net/publication/333677700_Machine_learning_for_email_spam_filtering_review_approaches_and_open_research_problems

- Fraud detection [2]
  https://www.kaggle.com/kabure/credit-card-fraud-prediction-rf-smote

- Predicting bank credit worthiness [3]
  https://www.researchgate.net/publication/326552481_Predictive_Modelling_for_Credit_Risk_Detection_using_Ensemble_Method

- Flower Classifier: more than 2 classes (**Multiclass Problem**) [4]
  https://www.geeksforgeeks.org/multiclass-classification-using-scikit-learn/

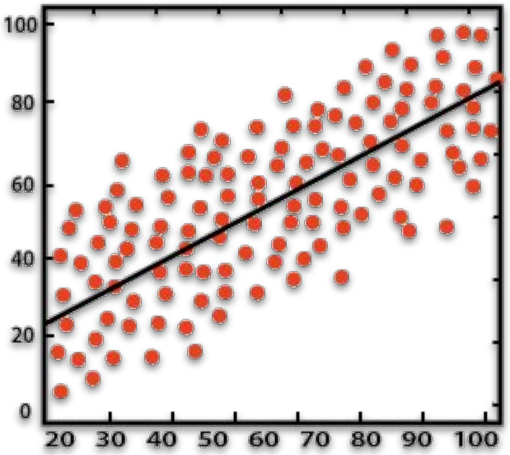


Classification

Classification is a process of **finding a function** which helps in dividing the dataset into **classes** based on **different parameters**.

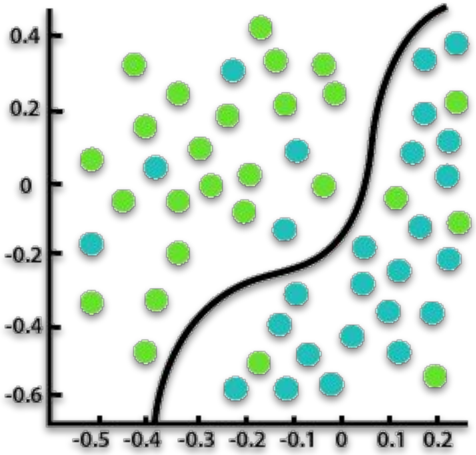
A classification algorithm must find the **mapping function** to map the **input(x)** to the discrete **output(y).**

Regression vs Classification



Regression

The output variable must be of continuous nature or real value.



Classification

In Classification, the output variable must be a discrete value.

Regression vs Classification

Regression

- In Regression, we try to find the best fit line (or curve), which can predict the output more accurately.
- A regression problem needs the prediction of a quantity.
- A regression problem containing multiple input variables is called a multivariate regression problem.

Classification

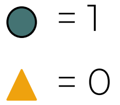
- In Classification, we try to find the decision boundary, which can divide the dataset into different classes.
- In a classification problem, data is labeled into one of two or more classes.
- A classification having problem with two classes is called binary classification, and more than two classes is called multi-class classification

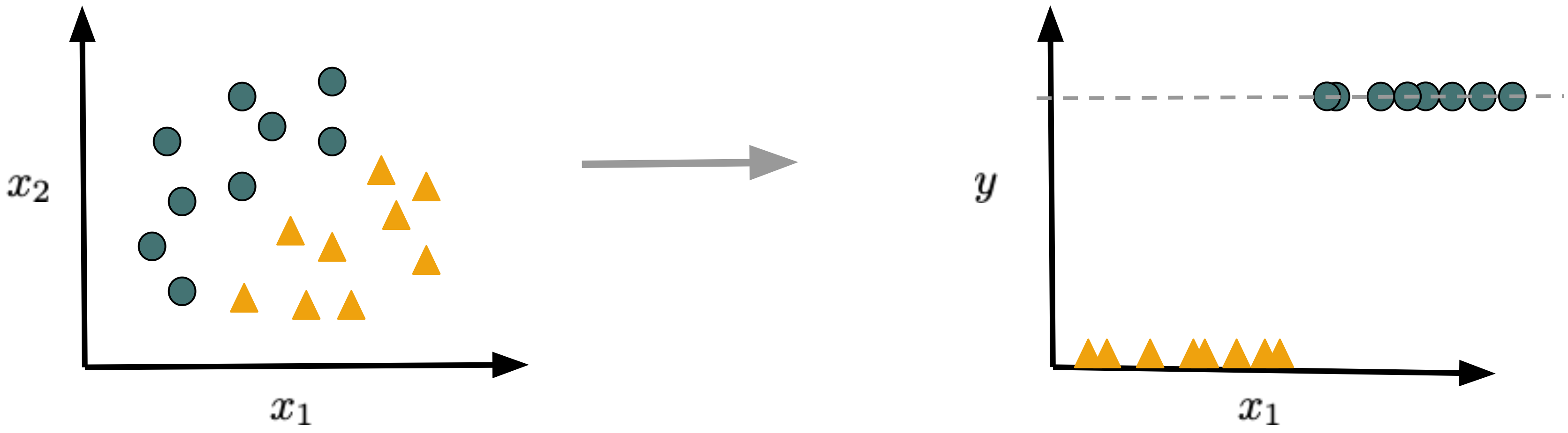
Regression vs Classification

Kahoot!

Logistic Regression

Logistic regression is the most common method for classifying data into discrete outcomes.

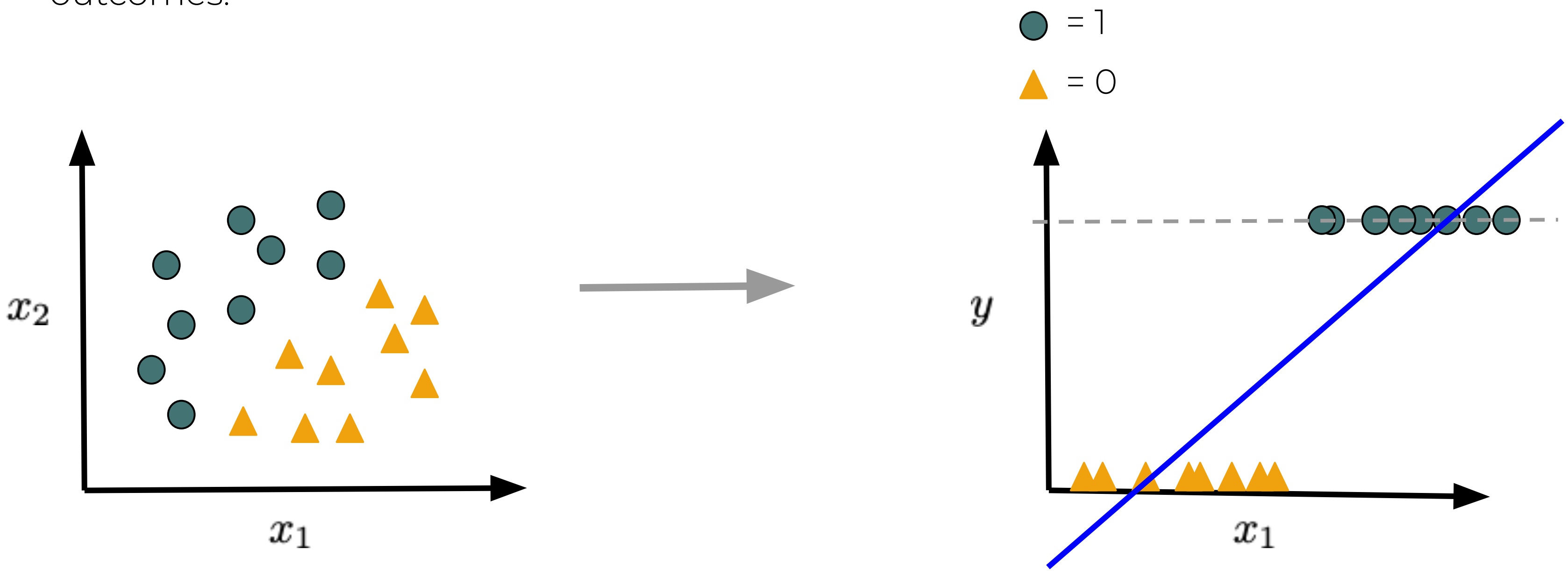




Note: In classification, the goal is to find that line (or curve) which separates the data at best

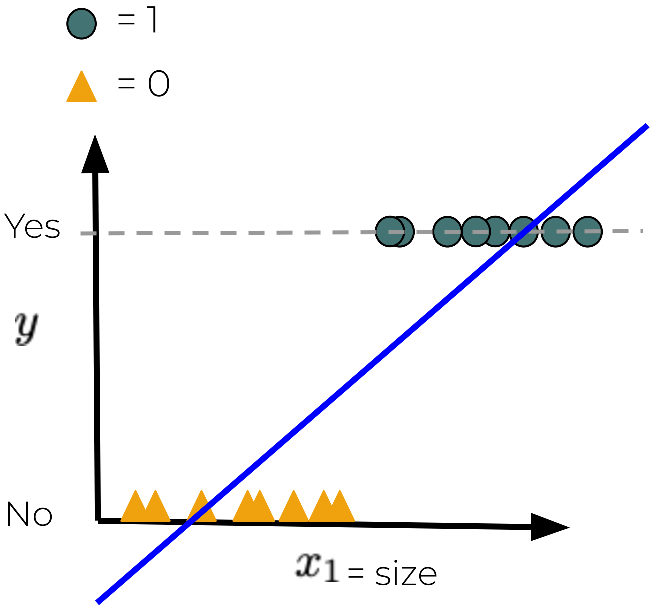
Logistic Regression

Logistic regression is the most common method for classifying data into discrete outcomes.



Note: In classification, the goal is to find that line (or curve) which separates the data at best

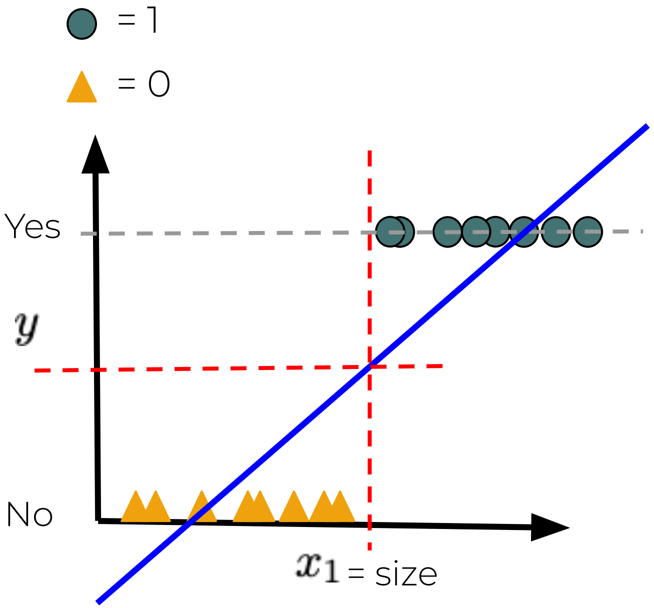
Logistic Regression



$$h_\theta(x) = \theta^T x$$

$$(y = \beta_0 + \beta_1 x)$$

Logistic Regression

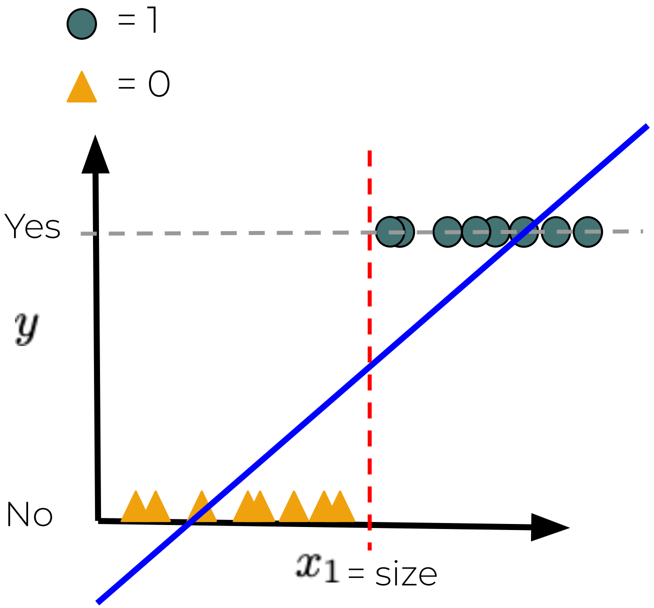


$$h_\theta(x) = \theta^T x$$

$$h_\theta(x) \geq 0.5 \rightarrow y = 1$$

$$h_\theta(x) < 0.5 \rightarrow y = 0$$

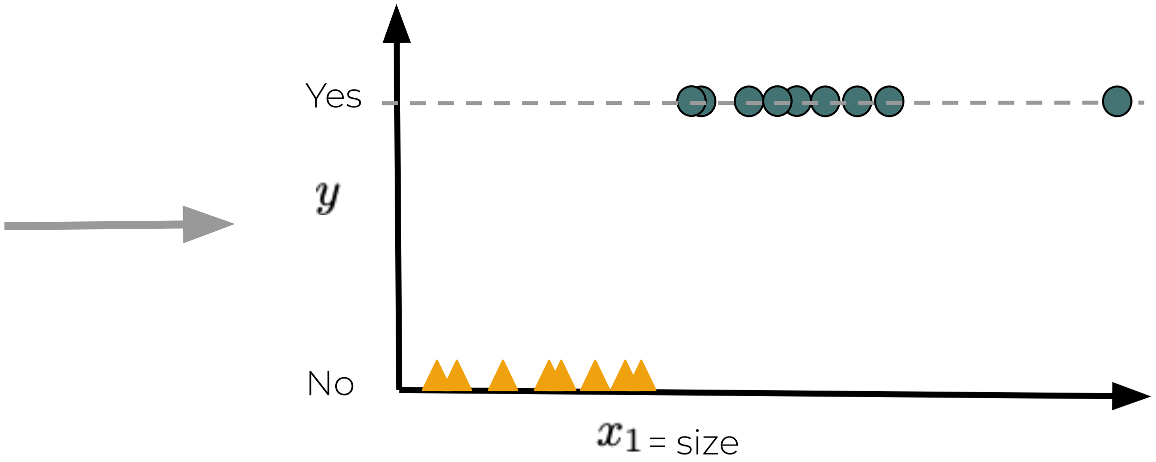
Logistic Regression



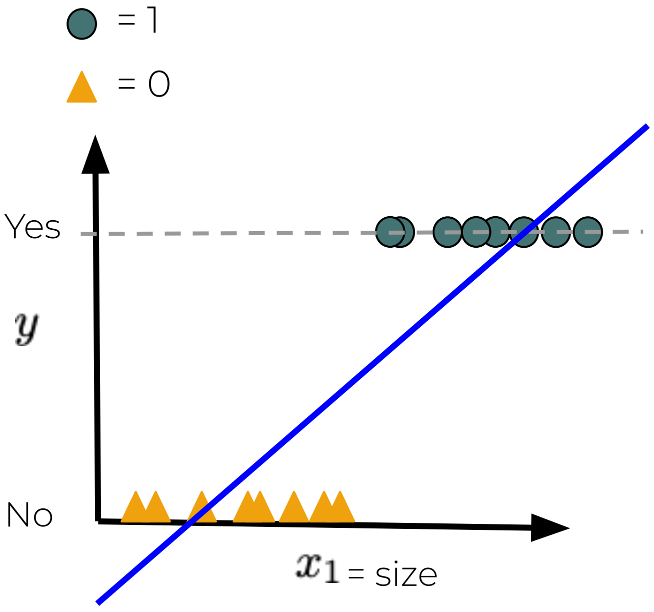
$$h_\theta(x) = \theta^T x$$

$$h_\theta(x) \geq 0.5 \rightarrow y = 1$$

$$h_\theta(x) < 0.5 \rightarrow y = 0$$



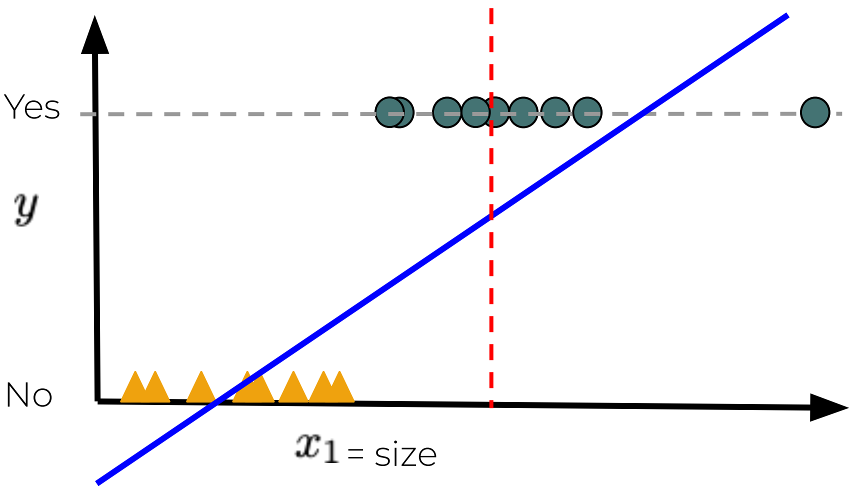
Logistic Regression



$$h_\theta(x) = \theta^T x$$

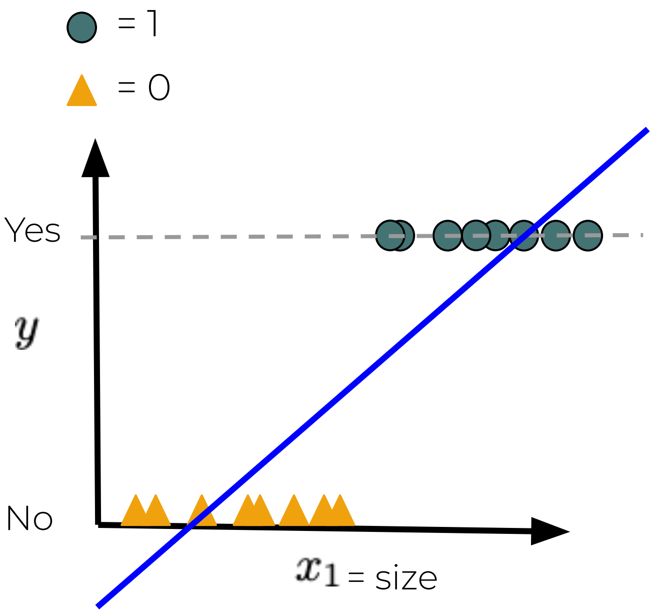
$h_\theta(x) \geq 0.5 \rightarrow y = 1$

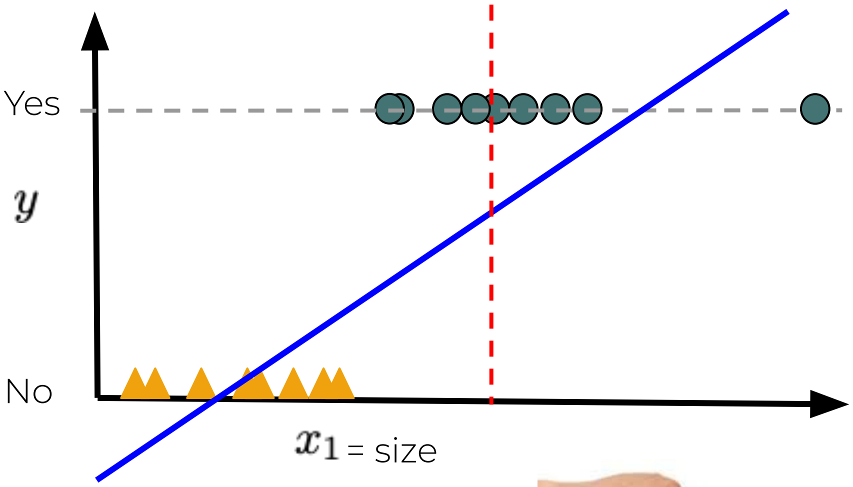
$h_\theta(x) < 0.5 \rightarrow y = 0$



IMMUNE CODING INSTITUTE

Logistic Regression



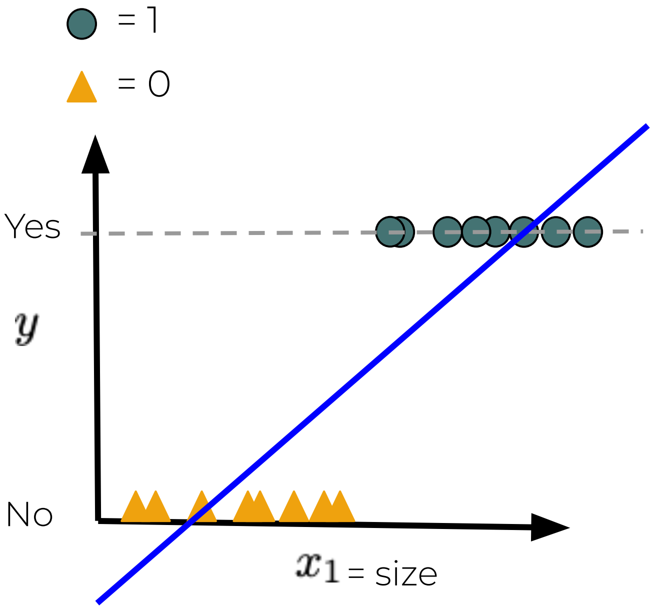


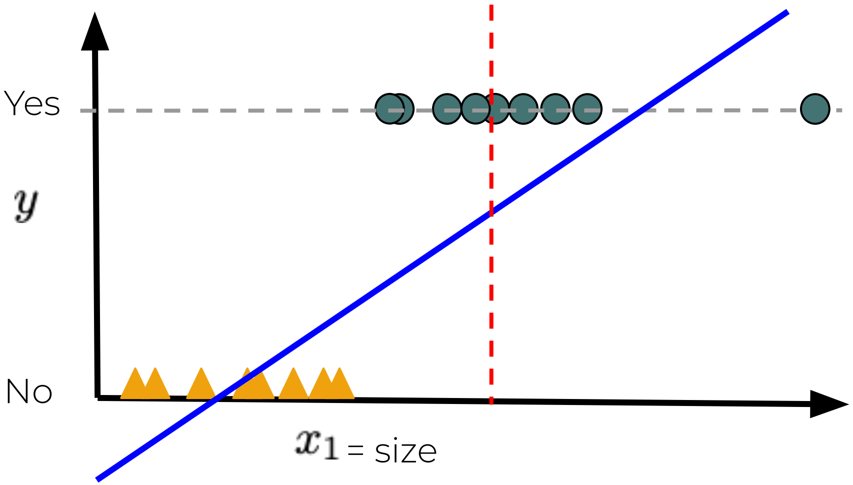
$$h_\theta(x) = \theta^T x$$

$$h_\theta(x) \geq 0.5 \rightarrow y = 1$$

$$h_\theta(x) < 0.5 \rightarrow y = 0$$

Logistic Regression





$$h_\theta(x) = \theta^T x$$

$$h_\theta(x) \geq 0.5 \rightarrow y = 1$$

$$h_\theta(x) < 0.5 \rightarrow y = 0$$

$$\boxed{0 \leq h_\theta(x) \leq 1}$$

Logistic Regression

Math

$$\boxed{0 \leq h_\theta(x) \leq 1}$$

$$y = \theta^T x$$

Logistic Regression

Math

$$\boxed{0 \leq h_\theta(x) \leq 1}$$

$$y = \theta^T x$$

$$g(y) = g(\theta^T x) = h_\theta(x)$$

Logistic Regression

Math

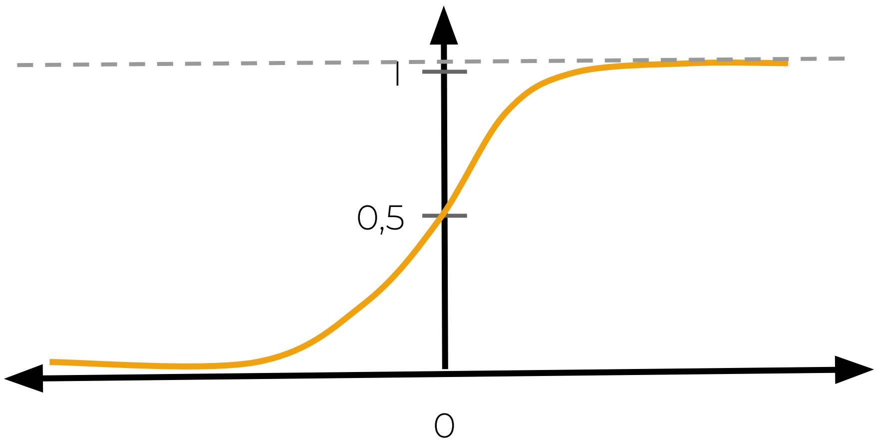
$$\boxed{0 \leq h_\theta(x) \leq 1}$$

$$y = \theta^T x$$

$$g(y) = g(\theta^T x) = h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Sigmoid function

$$\boxed{g(z) = \frac{1}{1 + e^{-z}}}$$



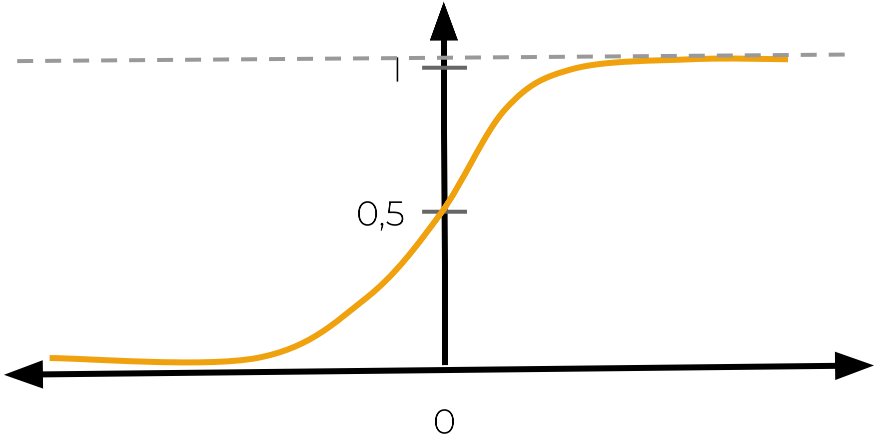
Logistic Regression

Math

$$\boxed{0 \le h_\theta(x) \le 1}$$

$$y = \theta^T x$$

$$g(y) = g(\theta^T x) = h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$



Sigmoid function

$$\boxed{g(z) = \frac{1}{1 + e^{-z}}}$$

Parameters: θ

$$threshold = th$$

Logistic Regression

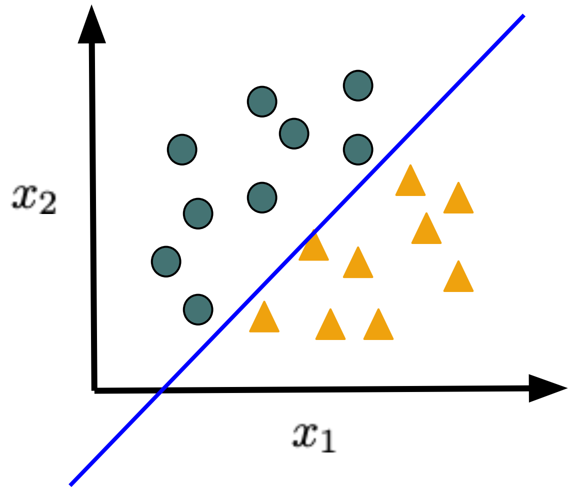
Math

$$\boxed{0 \leq h_\theta(x) \leq 1}$$

$h_\theta(x) =$ estimated probability that y = 1 on input x

$$h_\theta(x) = P(y = 1|x;\theta)$$

$$P(y = 0|x;\theta) = 1 - P(y = 1|x;\theta)$$



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

Logistic Regression

How to choose parameters θ?

Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Logistic Regression

How to choose parameters θ?

Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Logistic regression:

$$J(\theta) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

IMMUNE CODING INSTITUTE

Logistic Regression

How to choose parameters θ?

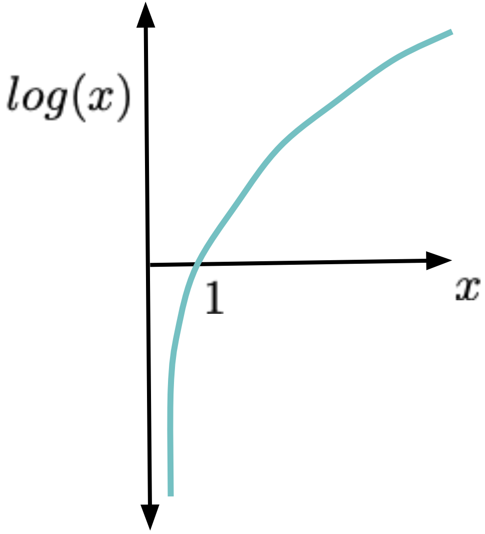
Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Logistic regression:

$$J(\theta) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



Logistic Regression

How to choose parameters θ?

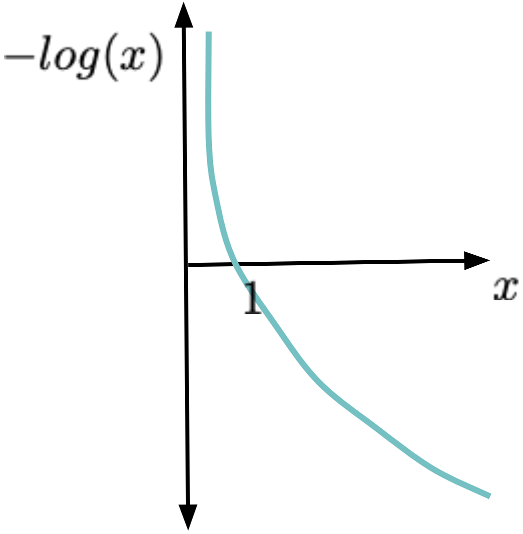
Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Logistic regression:

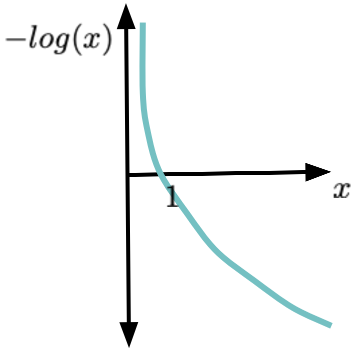
$$J(\theta) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



IMMUNE CODING INSTITUTE

Logistic Regression

How to choose parameters θ?



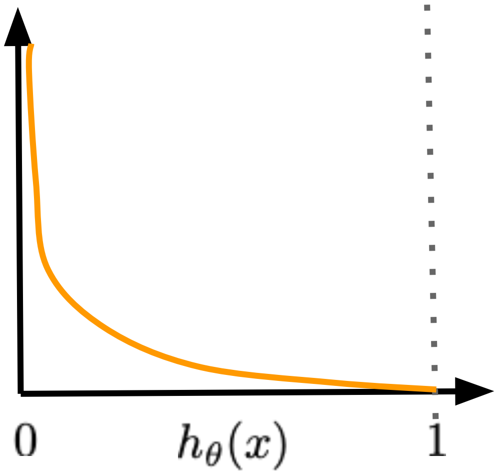
Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

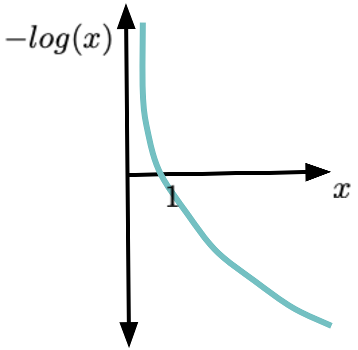
Logistic regression:

$$J(\theta) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



Logistic Regression

How to choose parameters θ?



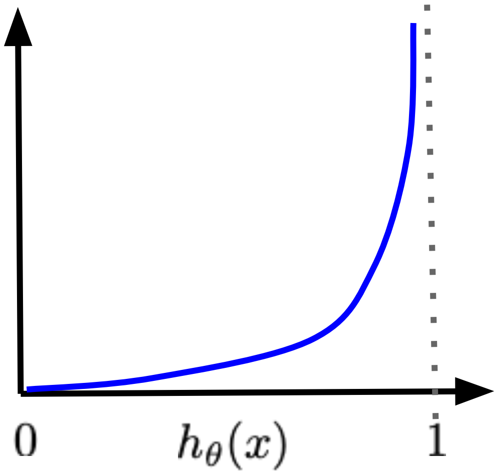
Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Logistic regression:

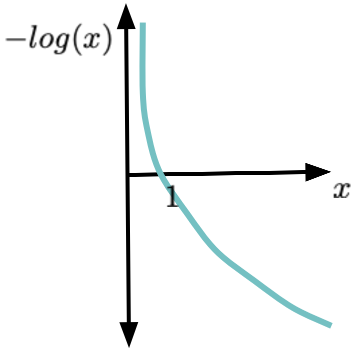
$$J(\theta) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



IMMUNE
CODING INSTITUTE

Logistic Regression

How to choose parameters θ?



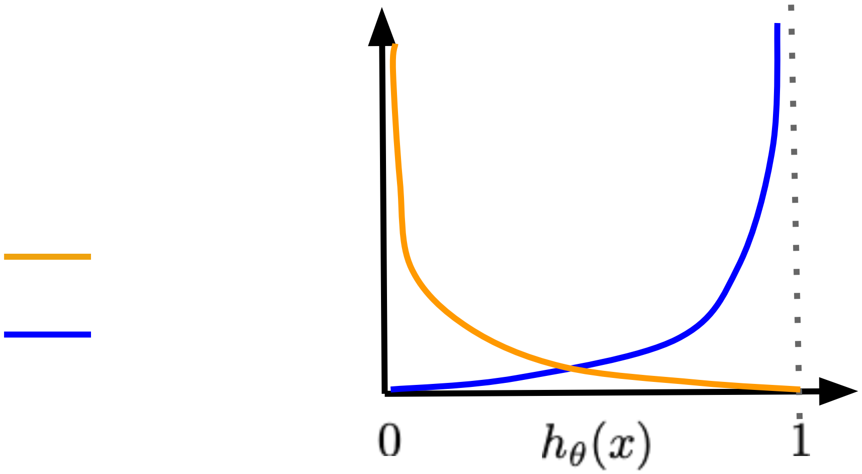
Cost function

Linear regression:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Logistic regression:

$$J(\theta) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



Classification Accuracy

Confusion Matrix

	Positive	Negative
Positive	True Positive	False Positive
Negative	False Negative	True Negative

Classification Accuracy

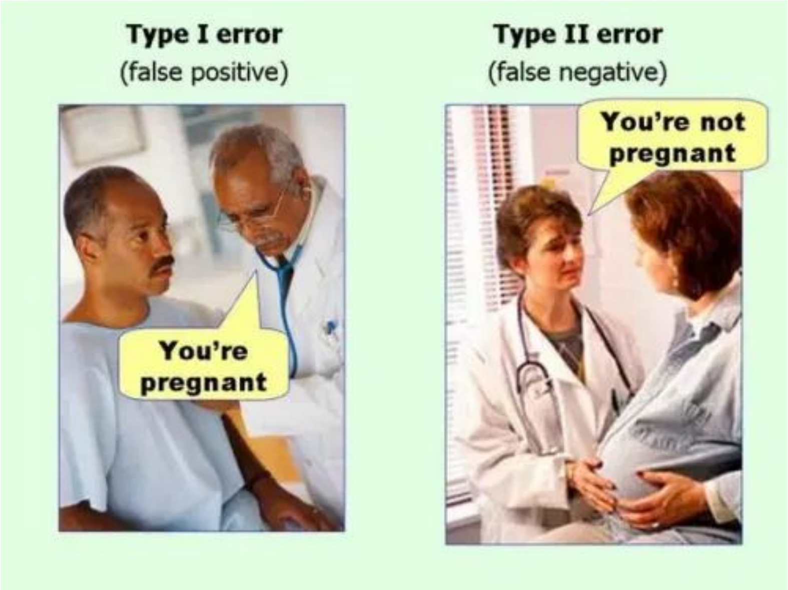
Confusion Matrix

	Positive	**Negative**
Positive	True Positive	False Positive
Negative	False Negative	True Negative

$$Sensitivity = \frac{TP}{TP + FN} = Recall$$

$$Specificity = \frac{TN}{TN + FP}$$

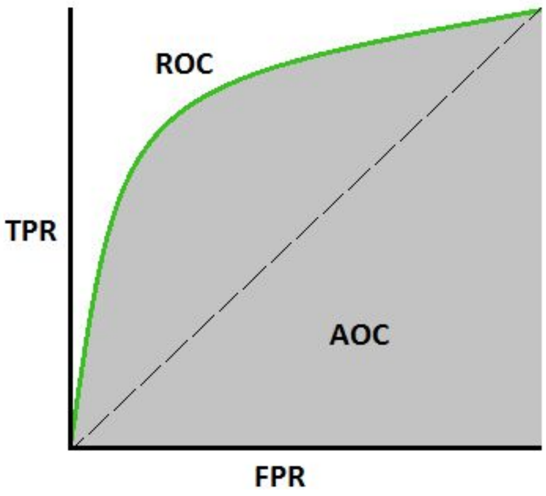
$$FPR = 1 - Specificity = \frac{FP}{TN + FP}$$



https://towardsdatascience.com/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124
https://en.wikipedia.org/wiki/Precision_and_recall

Classification Accuracy

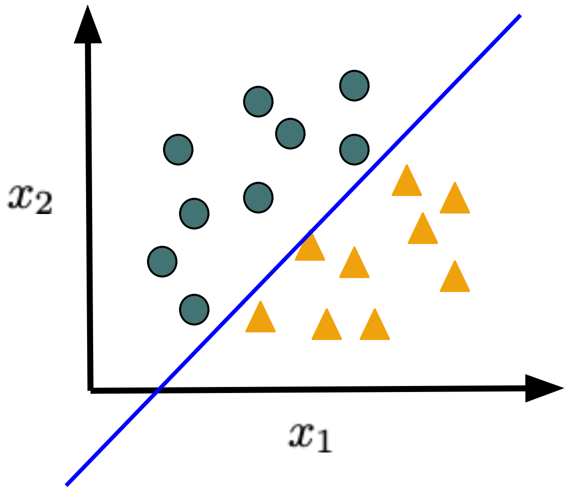
AUC - ROC Curve



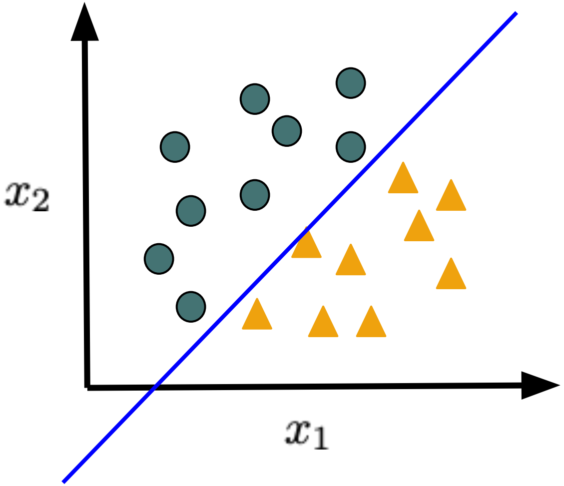
It tells how much model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting classes.

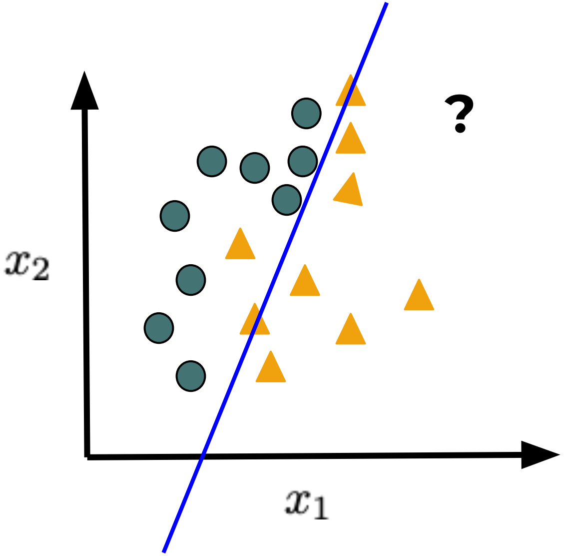
IMMUNE CODING INSTITUTE

Linear Separability

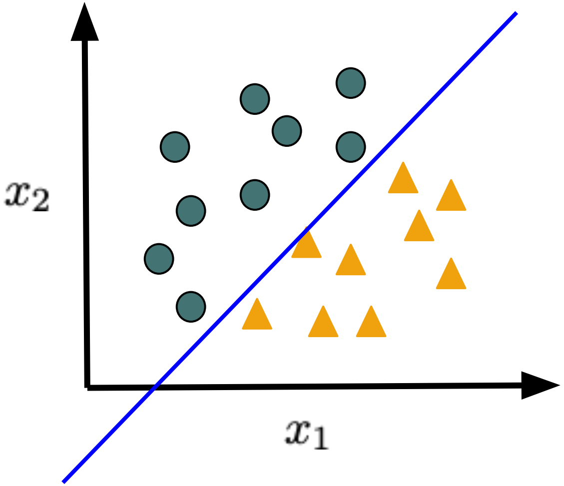


Linear Separability

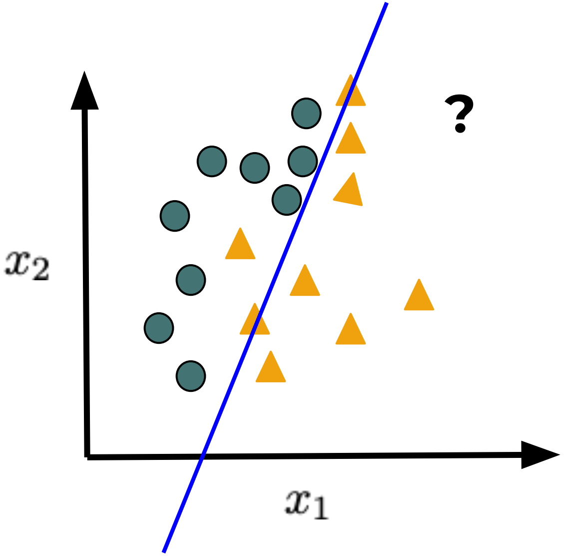


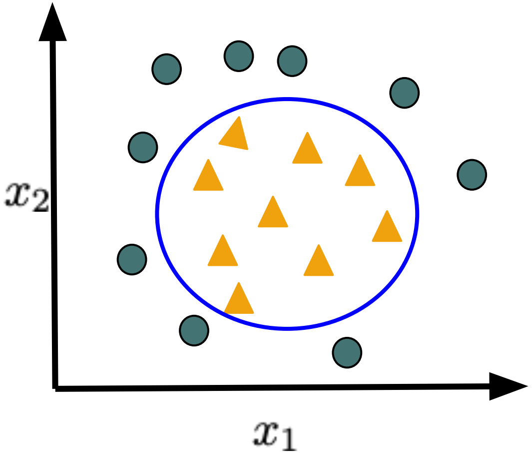


Linear Separability



Linearly separable

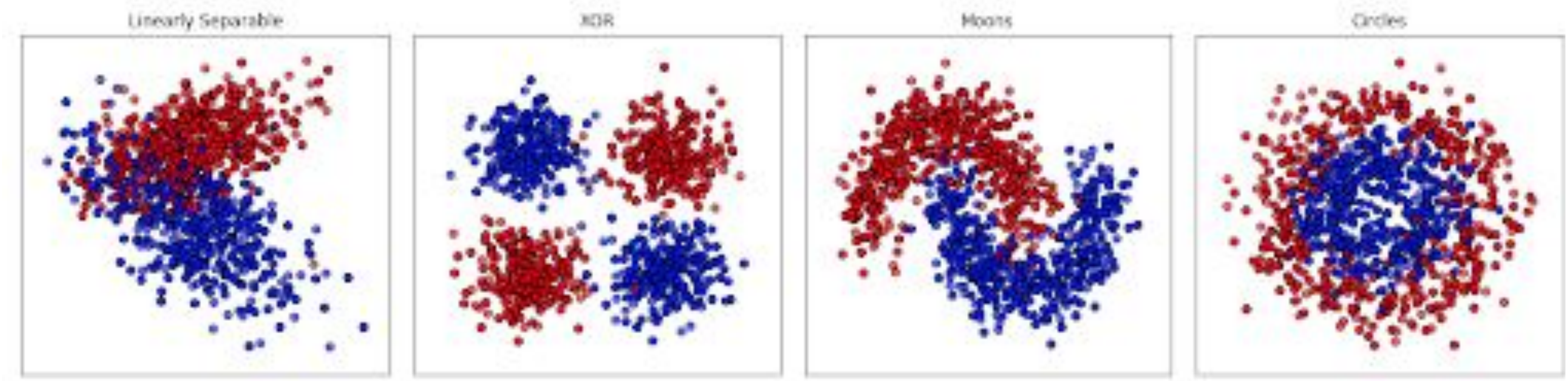




Linearly non-separable

IMMUNE
CODING INSTITUTE

Linear Separability



Other examples

Multiclass Classification

¿Hot dog or not Hog Dog?

VS

SILICON VALLEY

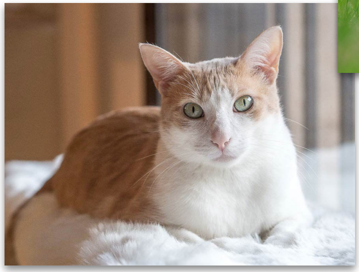
Multiclass Classification

Sometimes we will have to deal with classification problems with more than **two classes:**

- Medical diagnostic: Not ill, Cold, Flu

- Survey sentiment analysis: negative, neutral and positive.

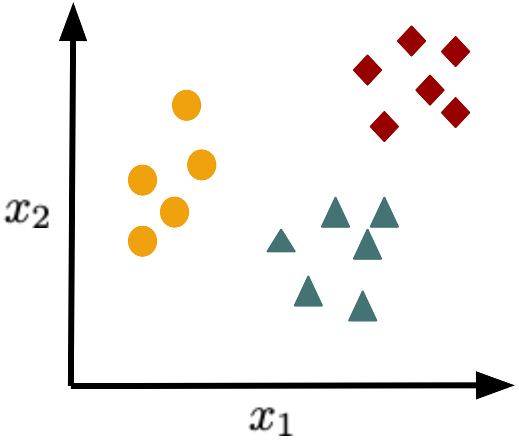
- Tagging images: dogs, cats and hamsters.



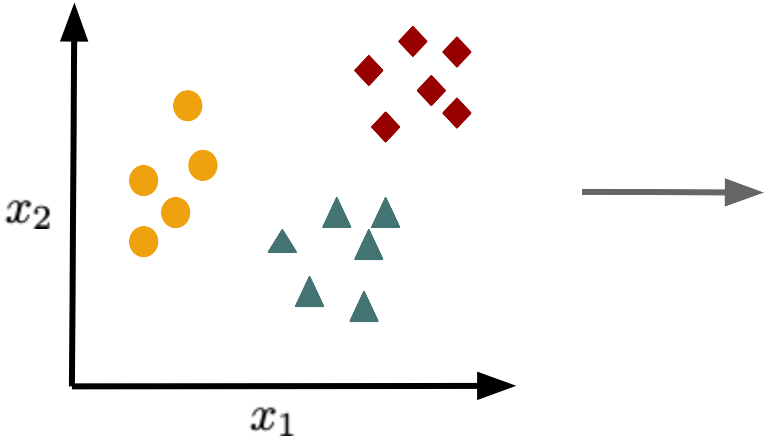
Multiclass Classification: One-vs-all

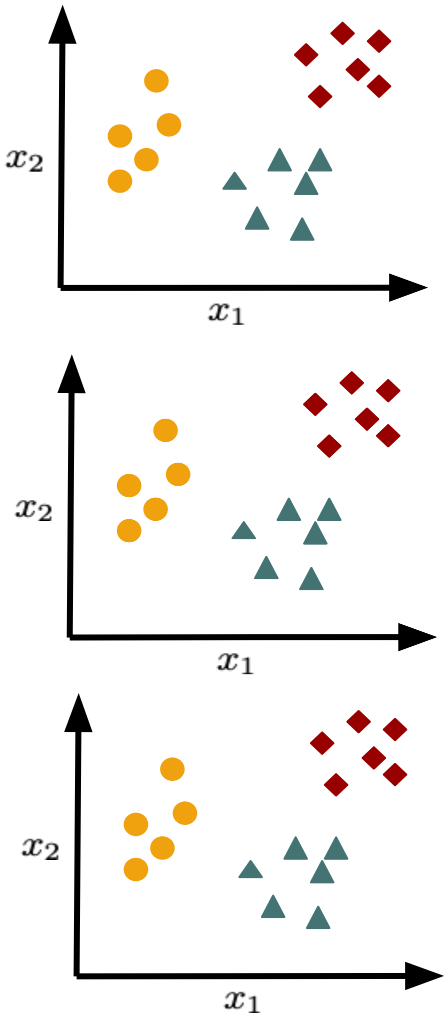
- Tagging images: dogs, cats and hamsters.

$y = 1$ $y = 2$ $y = 3$



One-vs-all

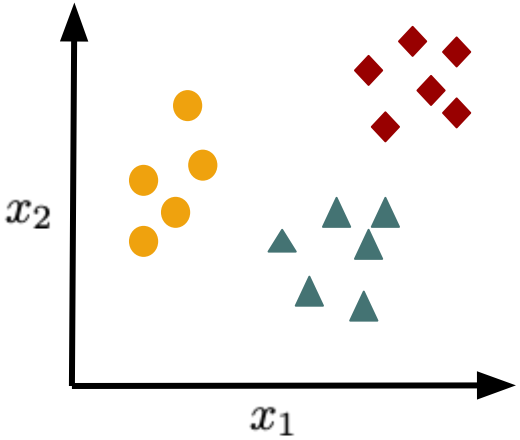


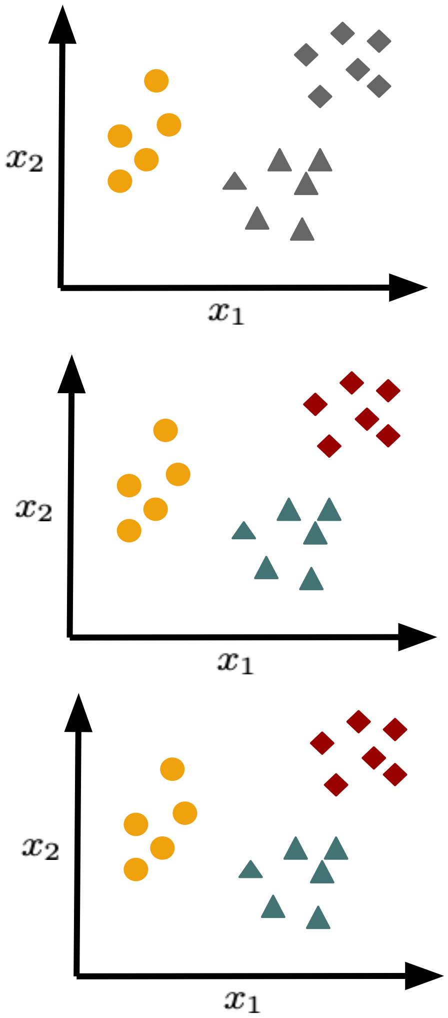


- Classes: dogs, cats and hamsters.

$y = 1 \qquad y = 2 \qquad y = 3$

One-vs-all

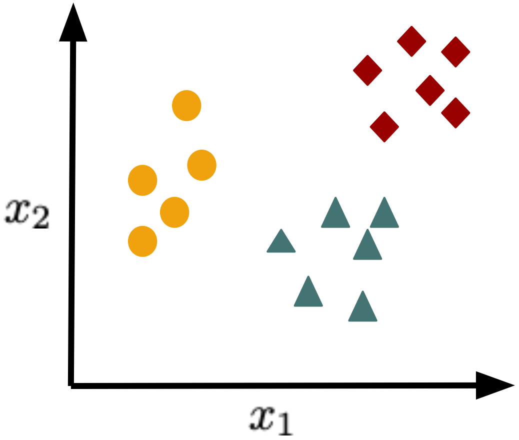


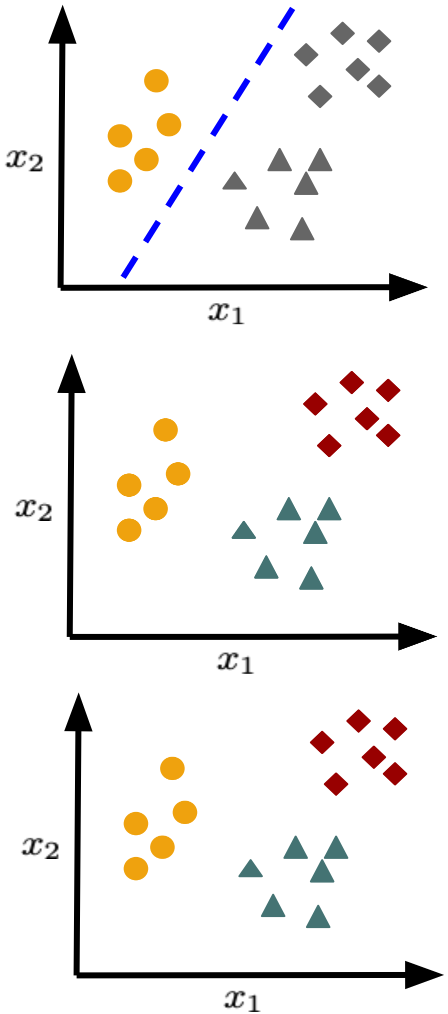


- Classes: dogs, cats and hamsters.

$y = 1$ $y = 2$ $y = 3$

One-vs-all



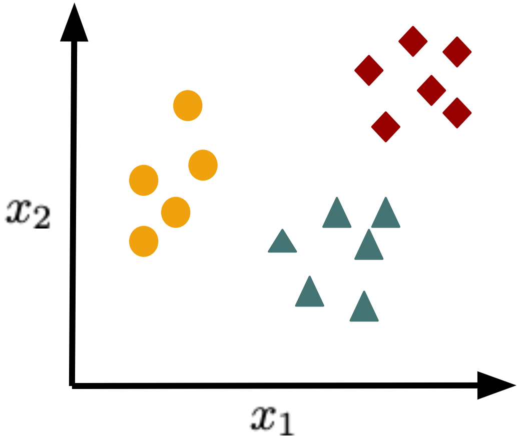


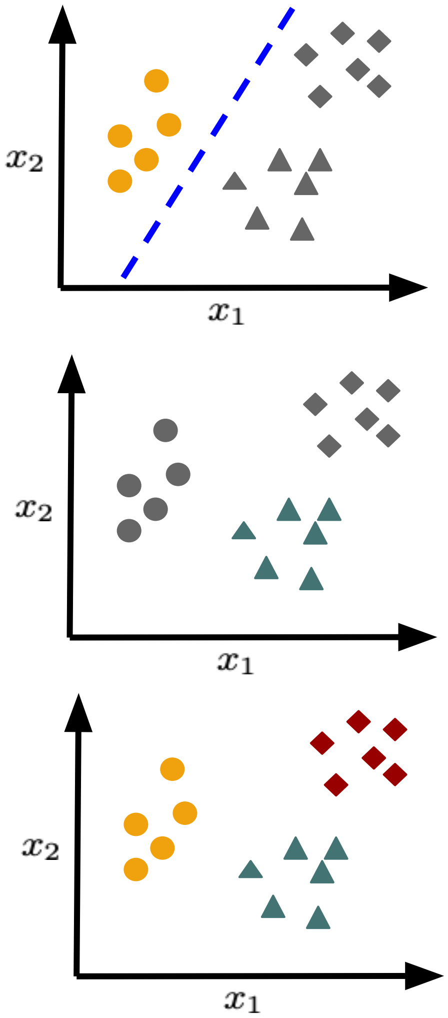
$h_\theta^{(1)}(x)$

- Classes: dogs, cats and hamsters.

$y = 1$ $y = 2$ $y = 3$

One-vs-all



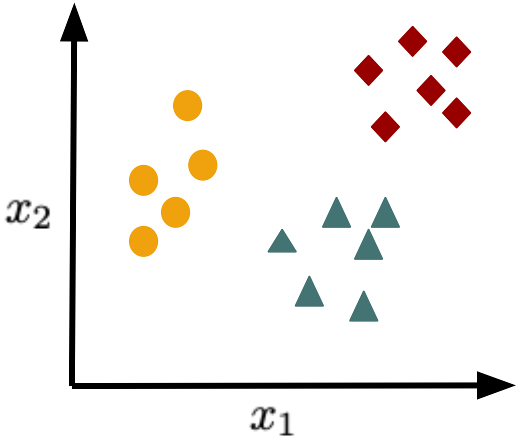


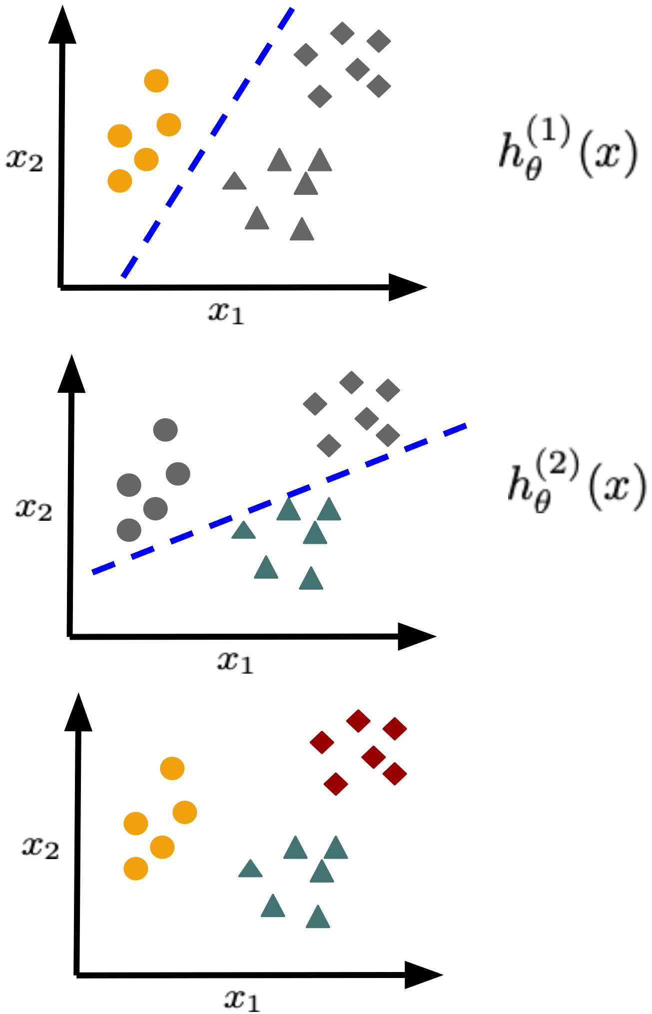
$h_\theta^{(1)}(x)$

- Classes: dogs, cats and hamsters.

$y = 1$ $y = 2$ $y = 3$

One-vs-all

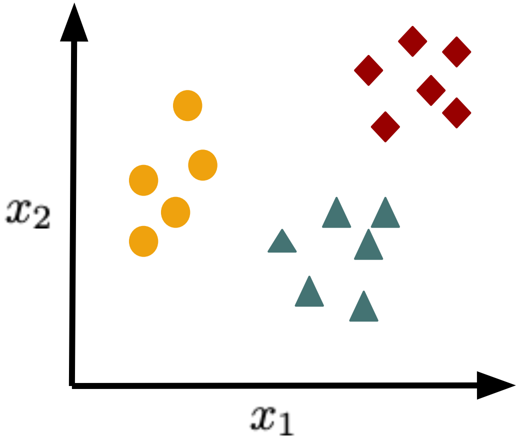


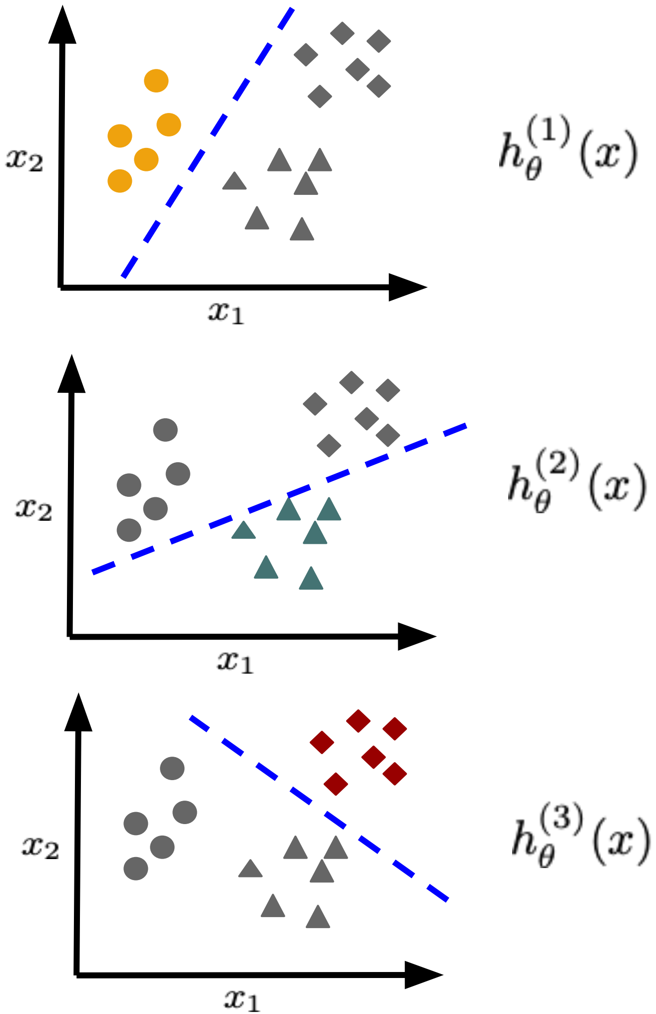


- Classes: dogs, cats and hamsters.

$y = 1$ $\quad$ $y = 2$ $\quad$ $y = 3$

One-vs-all





- Classes: dogs, cats and hamsters.

$y = 1$ $\quad$ $y = 2$ $\quad$ $y = 3$

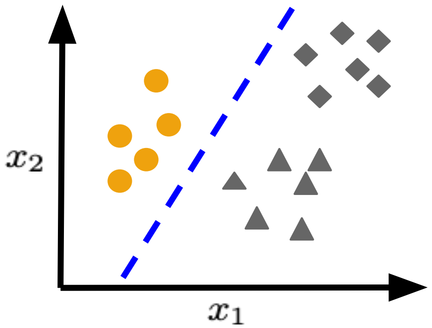
One-vs-all

Summary:

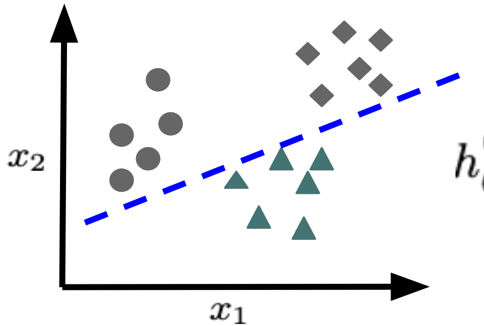
1. Train a logistic regression classifier for each class to predict the probability

2. On a new input *x*, to make a prediction, pick the class y that maximizes

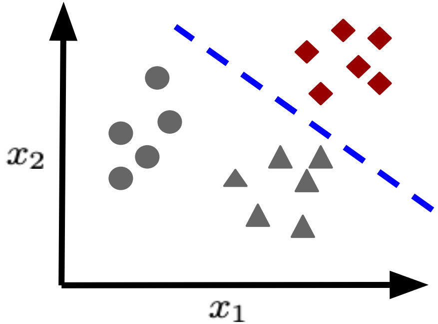
$$\max_i h_\theta^{(i)}(x)$$



$$h_\theta^{(1)}(x) = P(y = 1 | x, \theta)$$



$$h_\theta^{(2)}(x) = P(y = 2 | x, \theta)$$



$$h_\theta^{(3)}(x) = P(y = 3 | x, \theta)$$

One-vs-all

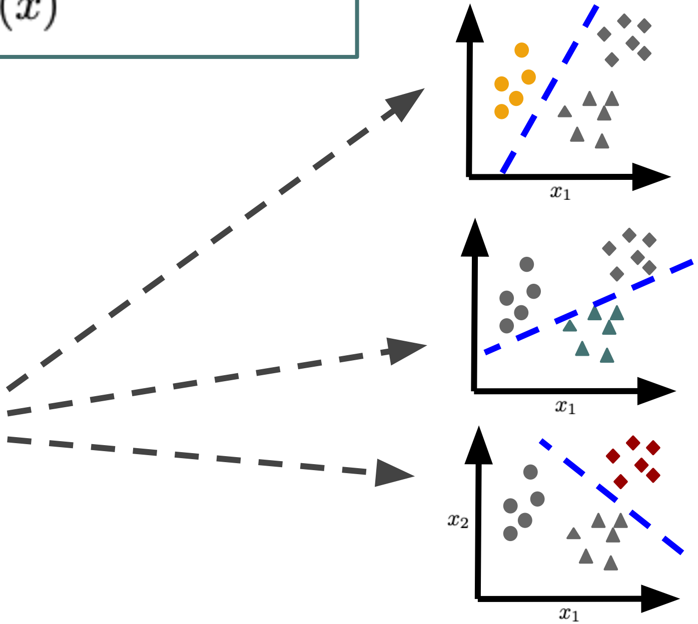
- Classes: dogs, cats and hamsters.

$$y = 1 \qquad y = 2 \qquad y = 3$$

On a new input *x*, to make a prediction, pick the class y that maximizes

$$\max_i h_\theta^{(i)}(x)$$

x



$$h_\theta^{(1)}(x) = P(y = 1|x, \theta) = 0.78$$

$$h_\theta^{(2)}(x) = P(y = 2|x, \theta) = 0.72$$

$$h_\theta^{(3)}(x) = P(y = 3|x, \theta) = 0.20$$

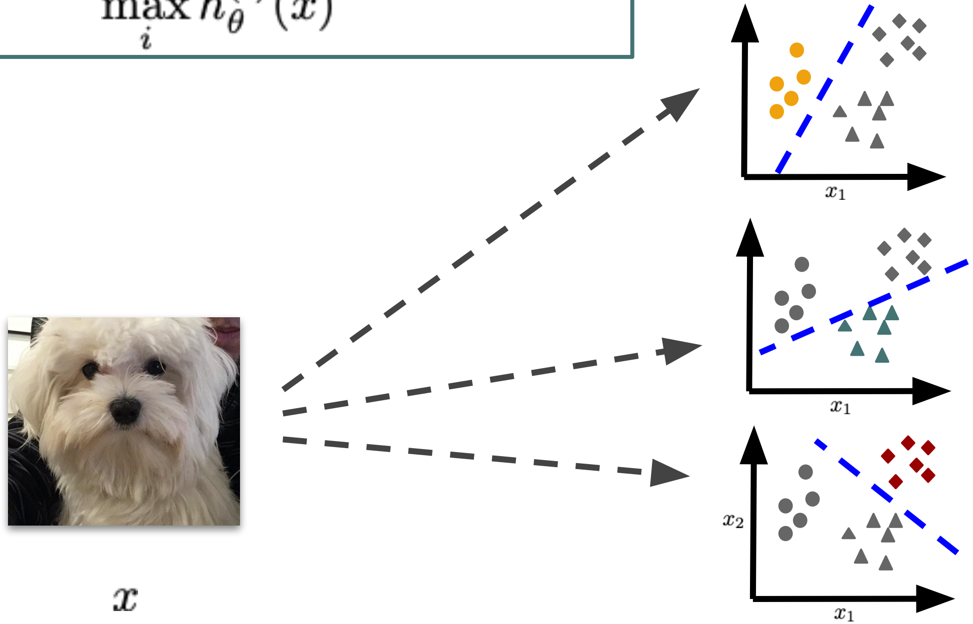
One-vs-all

- Classes: dogs, cats and hamsters.

$y = 1$ $\quad$ $y = 2$ $\quad$ $y = 3$

On a new input x, to make a prediction, pick the class y that maximizes

$$\max_i h_\theta^{(i)}(x)$$



$$h_\theta^{(1)}(x) = P(y = 1|x, \theta) = 0.78$$

$$h_\theta^{(2)}(x) = P(y = 2|x, \theta) = 0.72$$

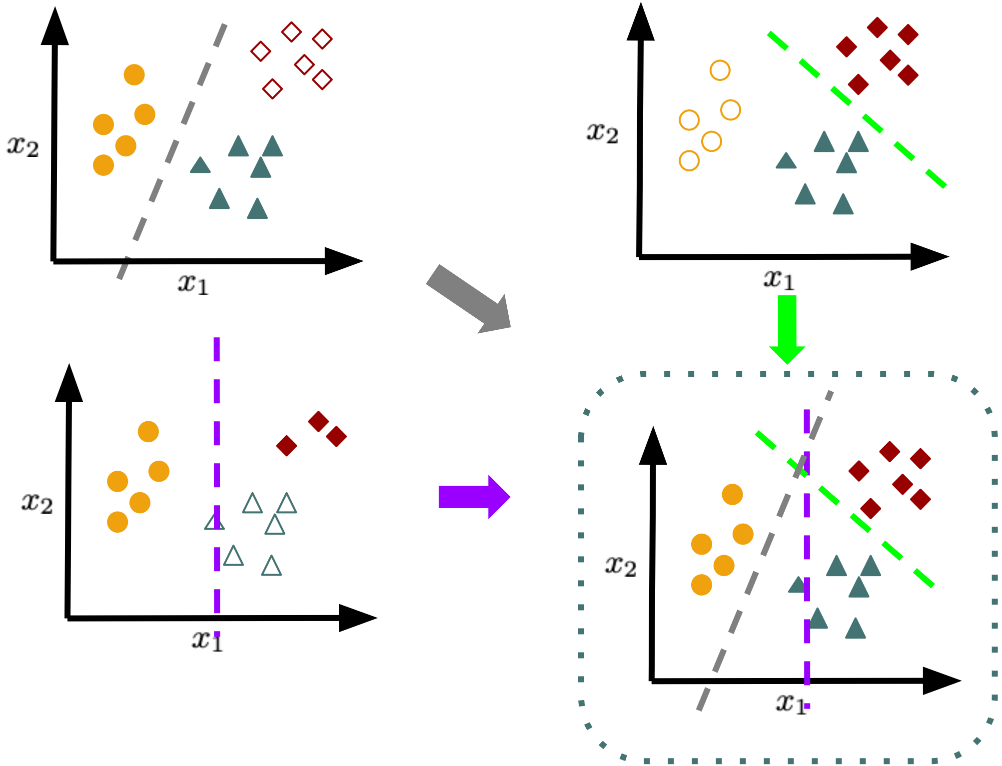
$$h_\theta^{(3)}(x) = P(y = 3|x, \theta) = 0.20$$

Others multi-class wrappers on binary classifiers

All vs All

Each binary classifier is trained to discriminate between individual pairs of classes and discard the rest.

Each new data point is evaluated by the classifier and assigned the class with the most votes.

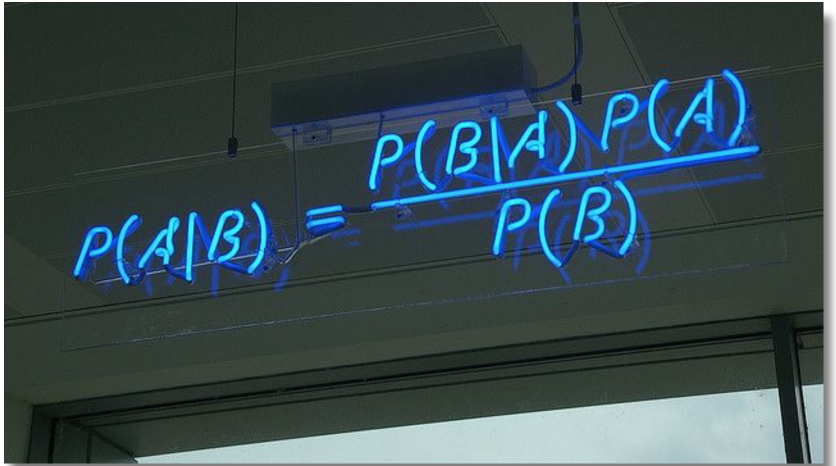


Naïve Bayes

Is a classification algorithm for binary and multiclass classification problems.

it is call Naive Bayes because the calculations of the probabilities for each class are simplified

The probabilities for each attribute are considered to be conditionally independent upon all other variables given a class.



$$P(Class_A|Data) = \frac{P(Data|Class_A)P(Class_A)}{P(Data)}$$

Naïve Bayes

$$P(Class_A|Data) = \frac{P(Data|Class_A)P(Class_A)}{P(Data)}$$

$$P(Class_1|X_1, X_2, \cdots, X_n) = P(X_1|Class_1) * P(X_2|Class_1) * \cdots * P(X_n|Class_1) * P(Class_1)/P(Data)$$

Naïve Bayes

$$P(Class_A|Data) = \frac{P(Data|Class_A)P(Class_A)}{P(Data)}$$

$$P(Class_1|X_1, X_2, \cdots, X_n) = P(X_1|Class_1) * P(X_2|Class_1) * \cdots * P(X_n|Class_1) * P(Class_1)/P(Data)$$

IMMUNE
CODING INSTITUTE

Naïve Bayes Basic Algorithm

1. Separate By Class
2. Summarize Dataset
3. Summarize Data by Class
4. Compute the Gaussian Probability Density Function
5. Compute Class Probabilities

IMMUNE CODING INSTITUTE

Naïve Bayes Considerations

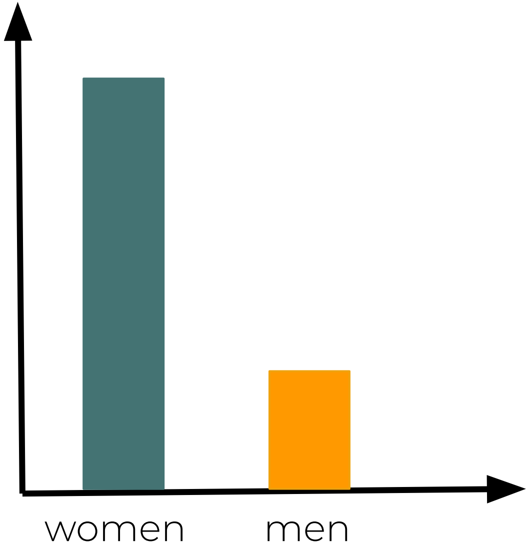
Pros

- Fast and relatively high accuracy in multiclass problem
- Perform well with less training data (assuming feature independence and categorical)

Cons

- Assumption of independent predictors is almost impossible in real life situations.
- Assumption of normally distributed input features, if it's continuous.
- If the categorical variable has a category in the test data but not in the train data, the probability of this category will be assigned zero and prediction is not possible

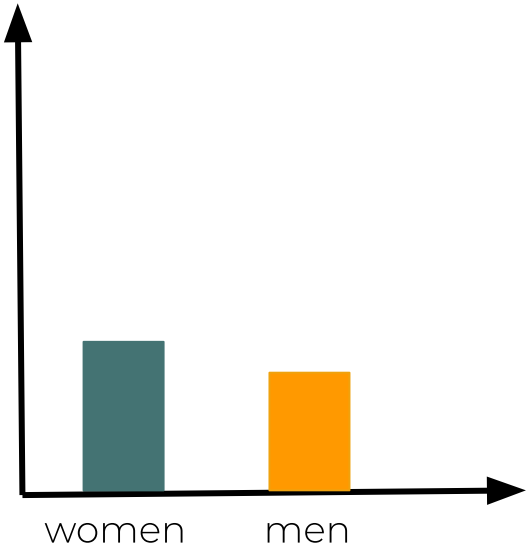
Imbalanced Data



Training sample

Imbalanced Data

1) Random Under-Sampling



Training sample

IMMUNE CODING INSTITUTE

Imbalanced Data

1) Random Under-Sampling

Pros

- Improve run time and storage

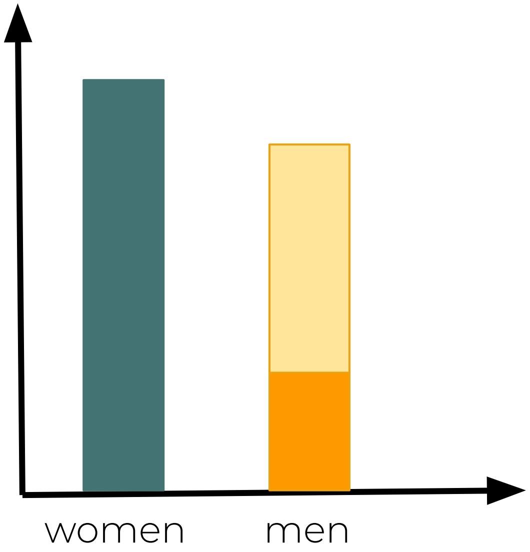
Cons

- It discard potentially useful information which could be important for building the classifier
- The sample chosen may be biased.

Training sample

Imbalanced Data

1) Random Over-Sampling



Training sample

IMMUNE
CODING INSTITUTE

Imbalanced Data

1) Random Over-Sampling

Pros

- Outperform under sampling
- Lead to no information loss

Cons

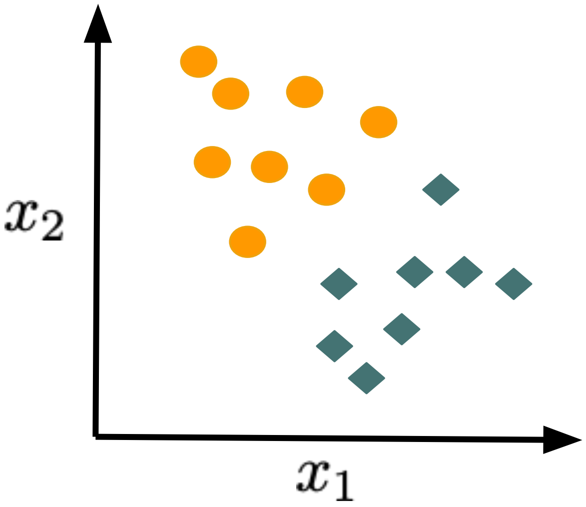
- It increases the likelihood of overfitting since it replicates the minority class events

Training sample

k-Nearest Neighbourhood (k-NN)

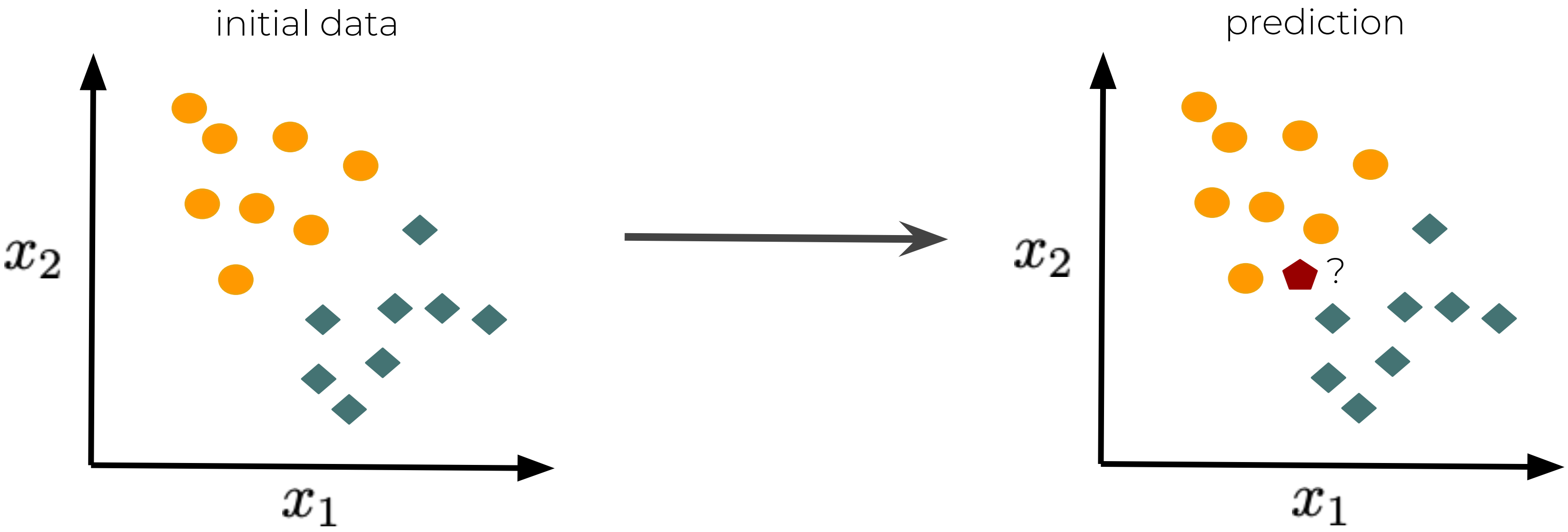
In a non-parametric method used for both classification and regression. It is considered a lazy learning method or, instance-based-learning, since does not need a training phase.

initial data



k-Nearest Neighbourhood (k-NN)

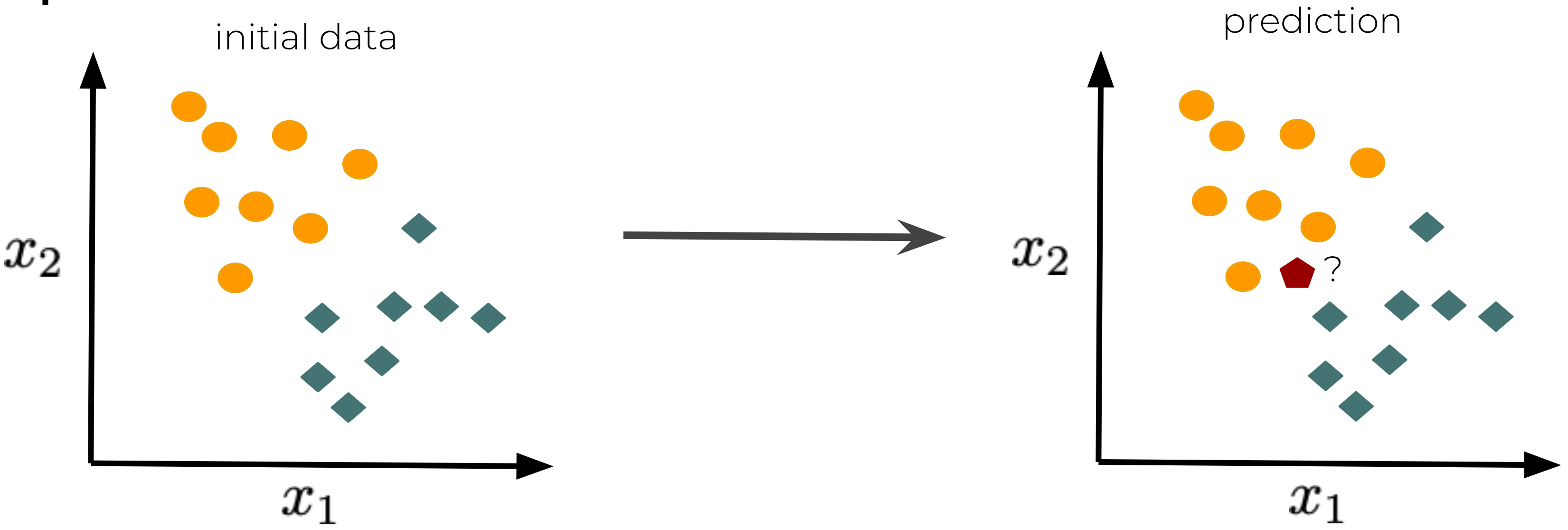
In a non-parametric method used for both classification and regression. It is considered a lazy learning method or, instance-based-learning, since does not need a training phase.



k-NN

$k \in \mathbb{N}$ is a constant (hyperparameter) defined by the user. Correspond to the number of "neighbors"

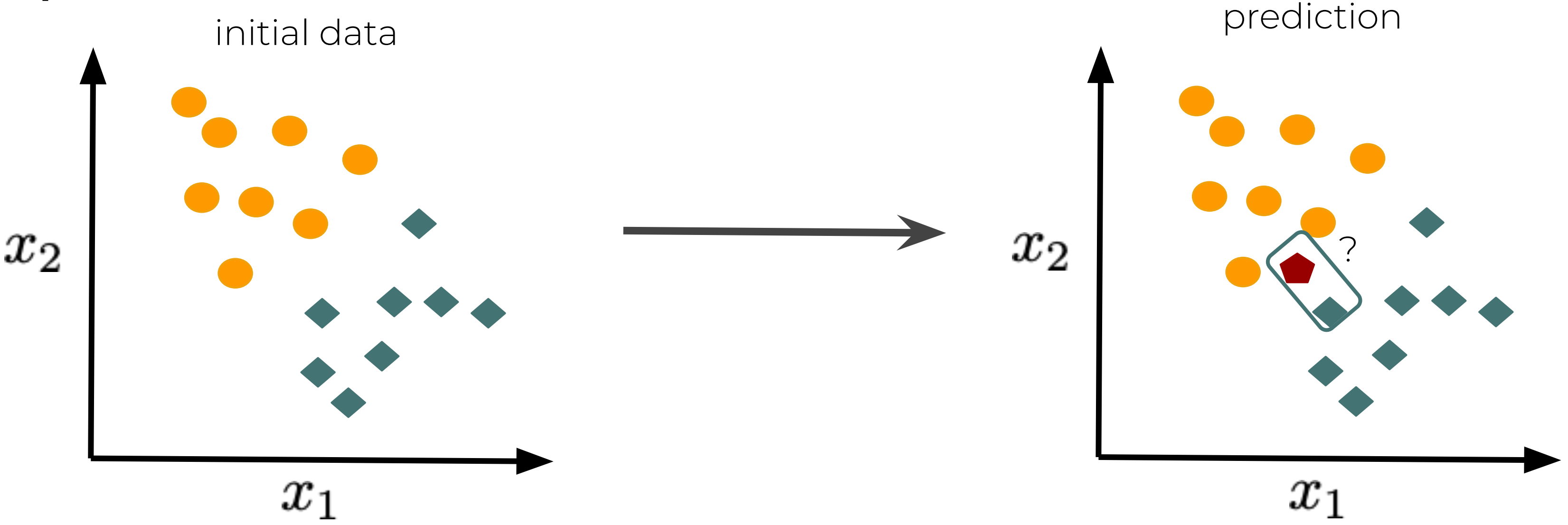
k **= 1**



k-NN

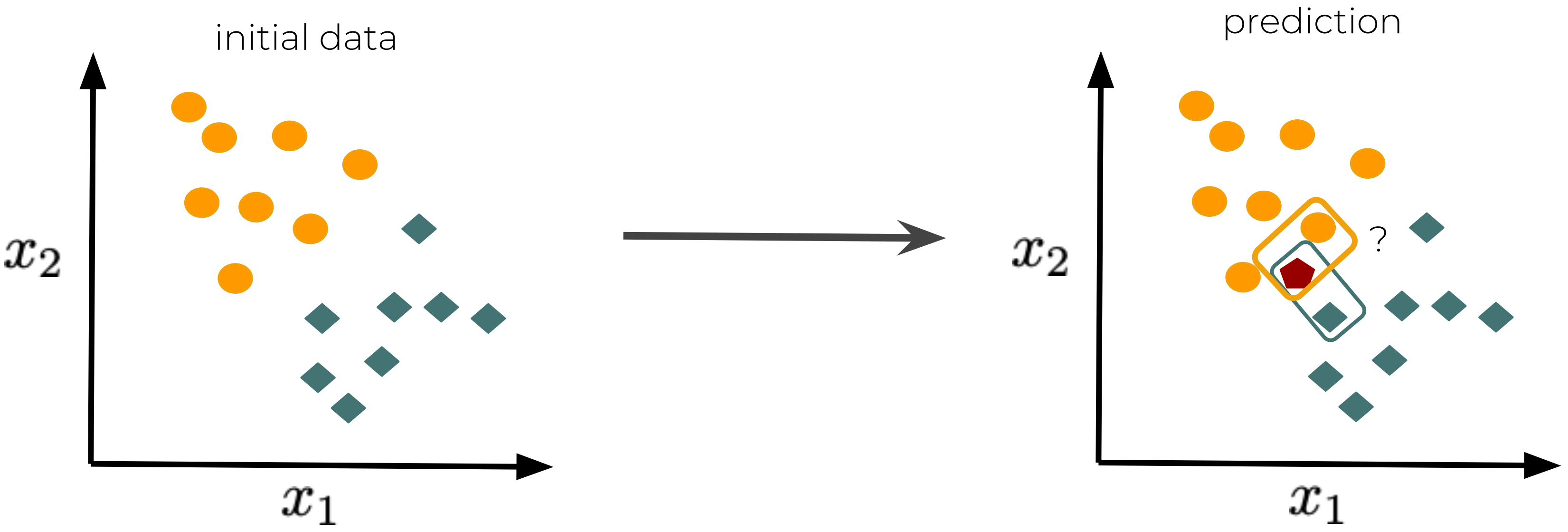
$k \in \mathbb{N}$ is a constant (hyperparameter) defined by the user. Correspond to the number of "neighbors"

***k* = 1**



k-NN

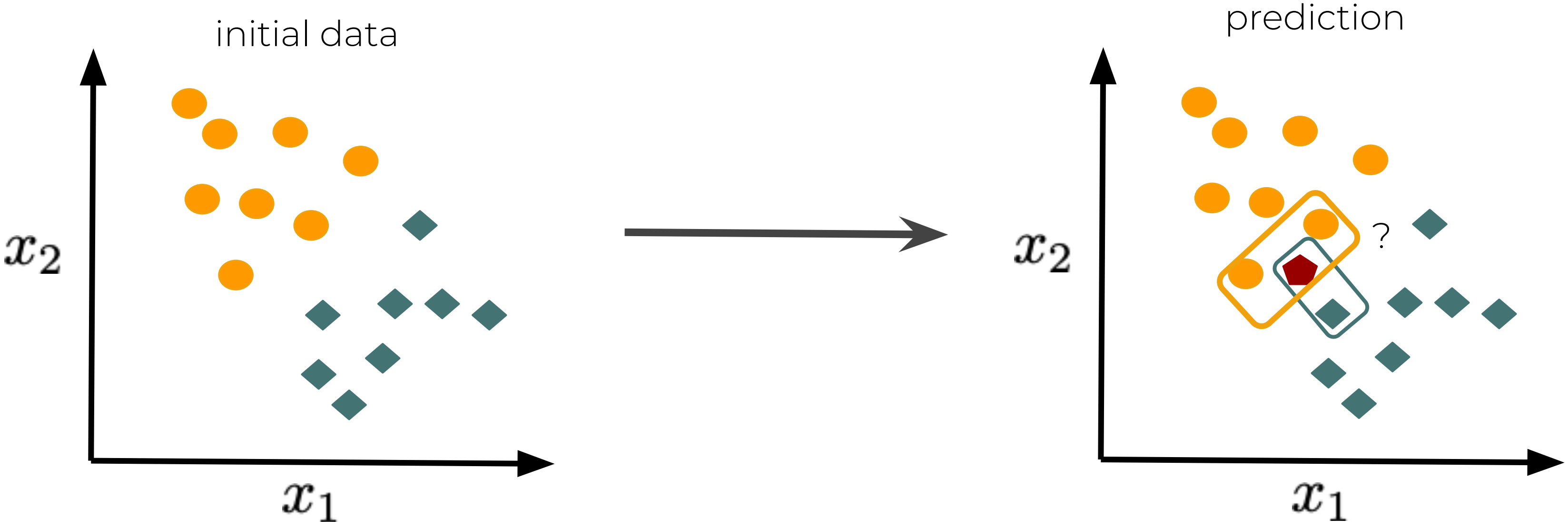
***k* = 2**



IMMUNE
CODING INSTITUTE

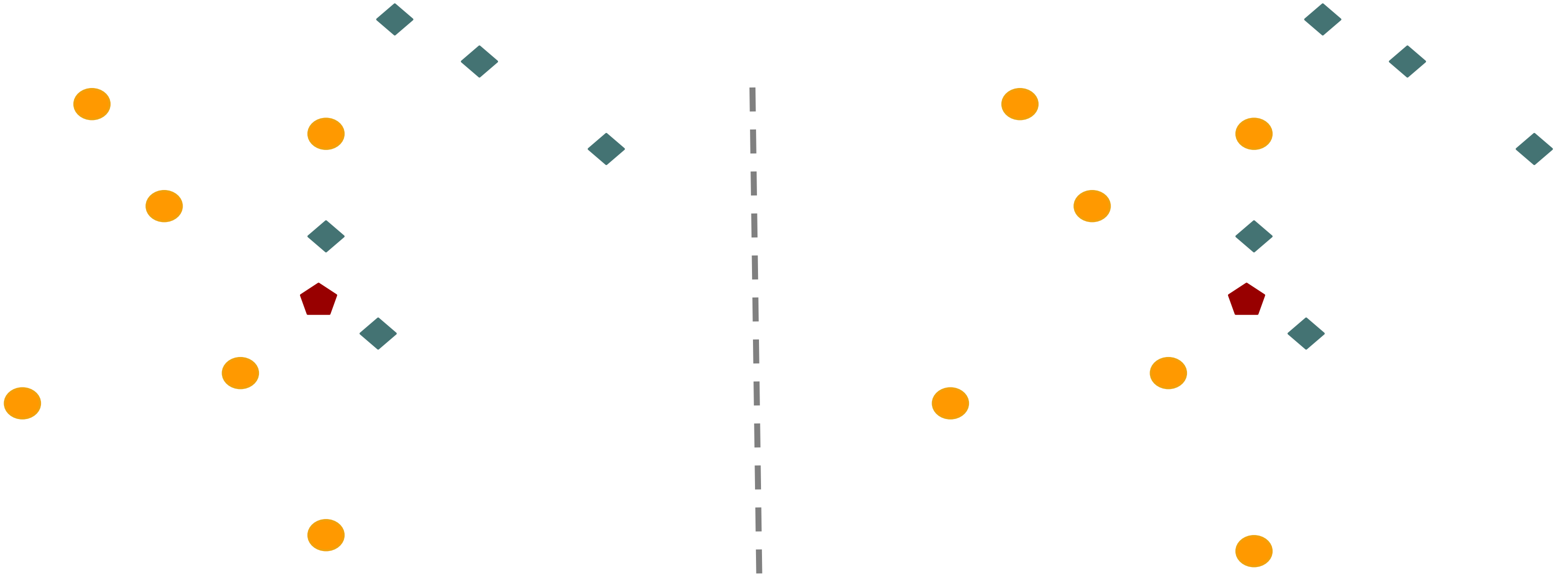
k-NN

***k* = 3**



k-NN

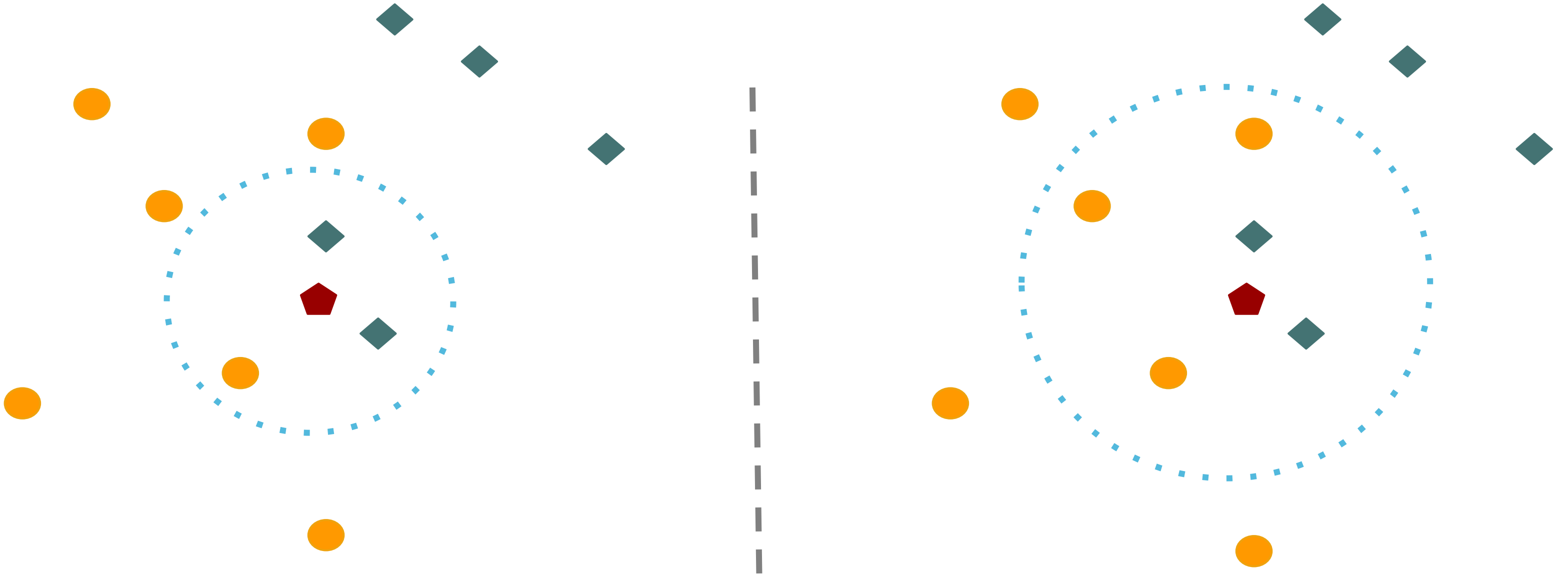
To prevent distant neighbors from having much influence, each neighbor can be made to vote in inverse proportion to the distance **1/d**.



k **= 3** ***k*** **= 5**

k-NN

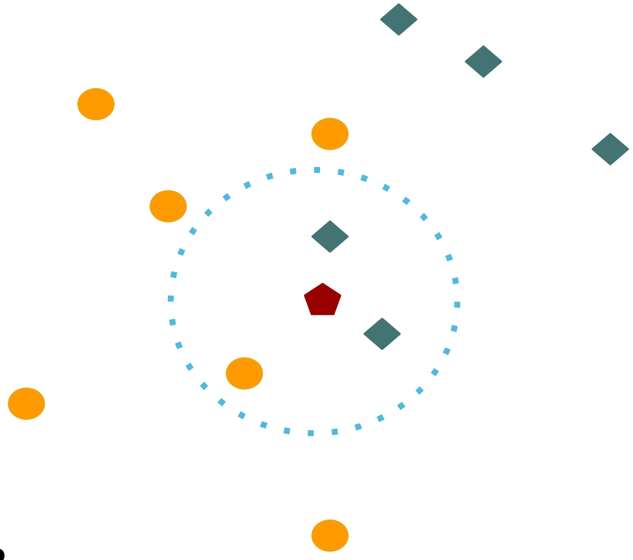
To prevent distant neighbors from having much influence, each neighbor can be made to vote in inverse proportion to the distance **1/d**.



***k* = 3**   ***k* = 5**

k-NN

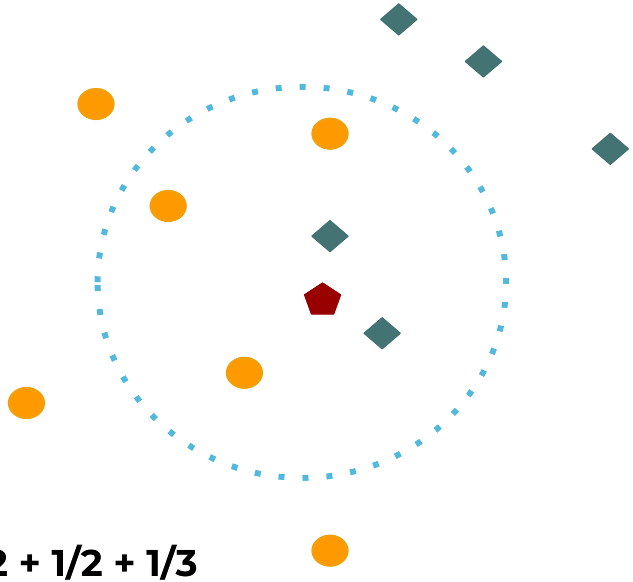
To prevent distant neighbors from having much influence, each neighbor can be made to vote in inverse proportion to the distance **1/d**.



***k* = 3**

= **1/2**

= **1/1 + 1/1 = 2**



***k* = 5**

= **1/2 + 1/2 + 1/3**

= **1/1 + 1/1 = 2**

k-NN: Considerations

- It is possible to used any metric to compute the distance ***d***. Normally the Euclidean distance is used:

$$d(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2}$$

- Attributes need to be standardized so that attributes with a high range do not have more than others (pre-processing)
  - Normalization $x'_{1j} = (x_{1j} - min_j)/(max_j - min_j)$
  - Standardization $x'_{1j} = (x_{1j} - \mu_j)/\sigma_j$

- If the attribute is nominal, the Hamming distance is used:

$$\sigma(x_{ia'}, x_{ia}) = \begin{cases} 1 & \text{if } x_{ia'} = x_{ia} \\ 0 & \text{otherwise} \end{cases}$$

IMMUNE CODING INSTITUTE

k-NN: Considerations

- It is possible to used any metric to compute the distance ***d***. Normally the Euclidean distance is used:

$$d(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2}$$

- Attributes need to be standardized so that attributes with a high range do not have more than others (pre-processing)
  - Normalization $x'_{1j} = (x_{1j} - min_j)/(max_j - min_j)$
  - Standardization $x'_{1j} = (x_{1j} - \mu_j)/\sigma_j$

- If the attribute is nominal, the Hamming distance is used:

$$\sigma(x_{ia'}, x_{ia}) = \begin{cases} 1 & \text{if } x_{ia'} = x_{ia} \\ 0 & \text{otherwise} \end{cases}$$

k-NN: Final considerations

- Very sensitive to irrelevant attributes and the curse of dimensionality
- As no model is built in *k*-NN, the boundary of separation is given directly by the training data
- Very slow, if there's a lot of training data
- It depends on the distance function
- Very sensitive to noise:
    - *k* = 1, the instances that are noise have a lot of influence.
    - If *k* is too high, you lose the idea of location.
- It can be also used for regression problems.

IMMUNE

CODING INSTITUTE