Les Mains à la pâte (suite)

Algorithme de détection de fruits (métalangage)

Pour i allant de 1 à Nb_fruits faire :

```
 Si (couleur[i]=="orange" ET diametre[i]>=7 ET diametre[i]<11)
alors :
    Afficher(« le fruit numéro », i , "est une orange »)
Sinon :
   Afficher(« le fruit numéro », i , « n’est pas une orange »)
Fin_si
```

Fin_Pour

Quels sont vos commentaires et autres remarques ?

Algorithme de détection de fruits (programmation python)

```
for i in range(len(dfr)) :

    if (dfr.couleur[i]=="orange" and dfr.diametre[i]>=7 and dfr.diametre[i]<11) :
        print("le fruit numéro », i+1 , "est une orange »)
    else :
        print(« le fruit numéro », i+1 , « n’est pas une orange »)
```

Quels sont vos commentaires et autres remarques ?

Algorithme de détection de fruits (métalangage)

```
Pour i allant de 1 à Nb_fruits faire :

    Si (couleur[i]=="orange" ET diametre[i]>=7 ET diametre[i]<11)
    alors :
        Afficher(« le fruit numéro », i , "est une orange »)
        predicted[i] = "orange"
    Sinon :
        Afficher(« le fruit numéro », i , « n'est pas une orange »)
        predicted[i] = "autre"
    Fin_si

Fin_Pour
```

Algorithme de détection de fruits (métalangage)

Quels sont vos commentaires et autres remarques ?

Pour i allant de 1 à Nb_fruits faire :

```
Si (couleur[i]=="orange" ET diametre[i]>=7 ET diametre[i]<11)
alors :
    Afficher(« le fruit numéro », i , "est une orange »)
    predicted[i] = "orange"
Sinon :
   Afficher(« le fruit numéro », i , « n'est pas une orange »)
    predicted[i] = "autre"
Fin_si
```

Fin_Pour

Algorithme de détection de fruits (programmation python)

On ajoute en 5ème colonne du dataframe une colonne nommée **predicted** et on initialise son contenu avec la modalité « **autre** » pour toutes les lignes

```
dfr.insert(4,'predicted',"autre")

for i in range(len(dfr)) :
```

```
if (dfr.couleur[i]=="orange" and dfr.diametre[i]>=7 and dfr.diametre[i]<11) :
    print(« le fruit numéro », str(i+1) , "est une orange »)
    dfr.predicted[i] = "orange"
else :
   print(« le fruit numéro », str(i+1) , « n'est pas une orange »)
   dfr.predicted[i] = "autre"
```

Quels sont vos commentaires et autres remarques ?

Algorithme de détection de fruits (programmation python)

On ajoute en 6ème colonne du dataframe une colonne nommée **actual**
et on initialise son contenu avec la modalité « **autre** » pour toutes les lignes

```
dfr.insert(5,‘actual',"autre")
```

```
for i in range(len(dfr)) :
```

```
if (dfr.fruit[i]=="orange") :  dfr.actual[i] = "orange"
```

Quels sont vos commentaires et autres remarques ?

Algorithme de détection de fruits (programmation python)

Vous l'aurez compris le cœur de l'algorithme va reposer sur

la comparaison des contenus de dfr.predicted et de dfr.fruits ou dfr.actual

dfr.fruits (ou dr.actual) : réalité/vérité

dfr.predicted : prédictions

dfr.predicted $\rightarrow$ dfr.actual ?

Quels sont vos commentaires et autres remarques ?

Algorithme de détection de fruits (programmation python)

Vous l'aurez compris le cœur de l'algorithme va reposer sur

la comparaison des contenus de dfr.predicted et de dfr.fruits ou dfr.actual

dfr.fruits (ou dr.actual) : réalité/vérité

dfr.predicted : prédictions

dfr.predicted → dfr.actual ?

Matrice de confusion

Quels sont vos commentaires et autres remarques ?

Matrice de confusion

Matrice de Confusion		Réalité	
		positive(+,1)	négatif(-,0)
Prédiction	positive(+,1)	VP	FP
	négatif(-,0)	FN	VN

$$Accuracy = \frac{VP+VN}{VP+VN+FP+FN}$$

(proportion de bonne prédictions sur l'ensemble des prédictions)

Matrice de Confusion		Etat de la personne	
		Malade	Non malade
Résultat Test	Test positif	VP	FP
	Test négatif	FN	VN

Matrice de Confusion		Nature du fruit	
		Orange	Autre
Prédiction	Orange	VP	FP
	Autre	FN	VN

5 paramètres pour évaluer la performance de l'algorithme

$$Accuracy = \frac{VP+VN}{VP+VN+FP+FN}$$

(proportion de bonne prédictions sur l'ensemble des prédictions)

$$Sensibilité\ (Recall) = \frac{VP}{VP+FN}$$

(proportion de positifs correctement prédits sur l'ensemble des positifs)

$$Spécificité\ (Précision) = \frac{VN}{VN+FP}$$

(proportion de négatifs correctement prédits
sur l'ensemble des négatifs)

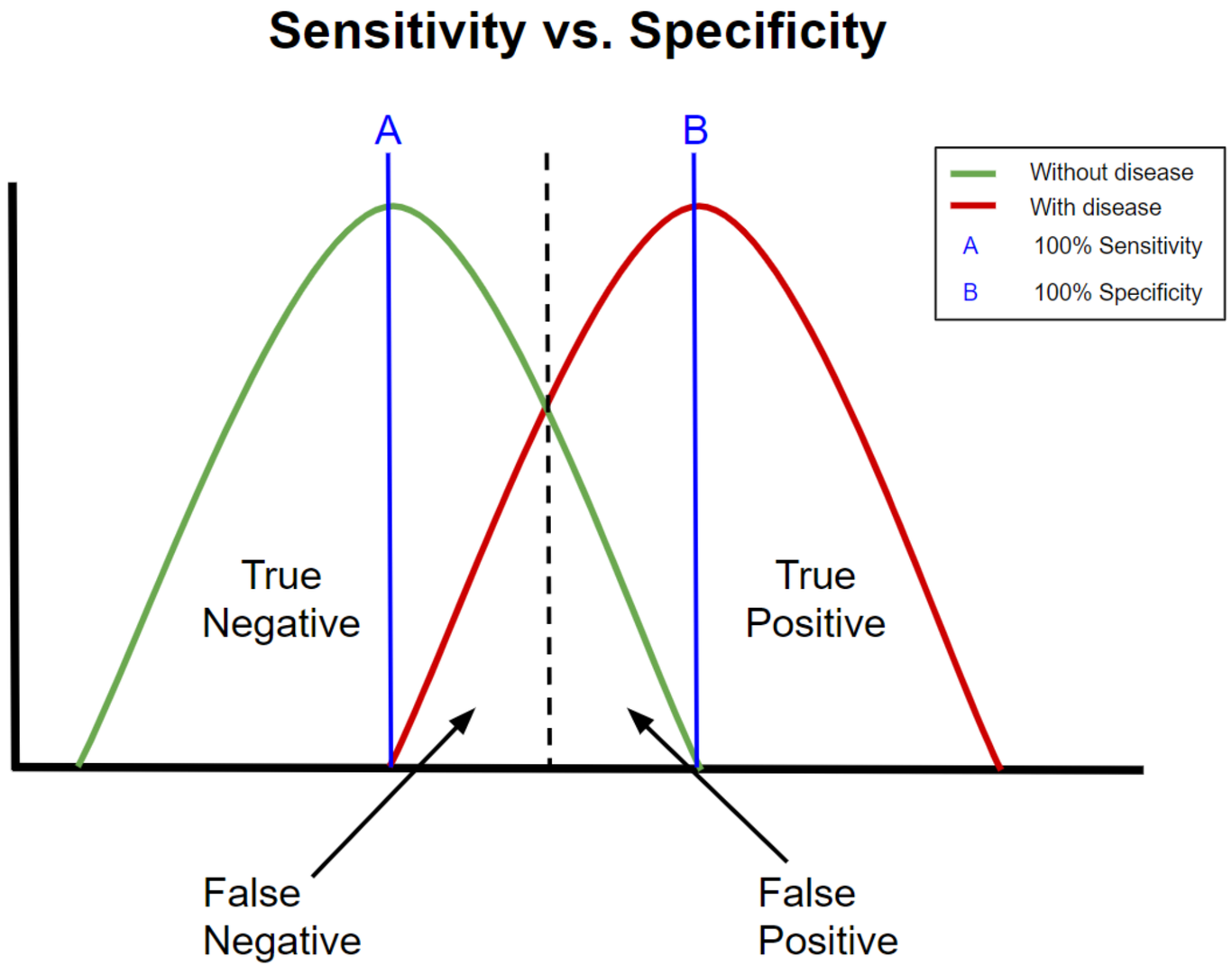
$$\text{VPP} = \frac{VP}{VP+FP}$$

(Valeur Prédictive Positive :
proportion de positifs correctement prédits sur l'ensemble des prédits positifs)

$$\text{VPN} = \frac{VN}{VN+FN}$$

(Valeur Prédictive Négative :
proportion de négatifs correctement prédits sur l'ensemble des prédits négatifs)

Un peu d'épidémiologie : Sensibilité vs. Spécificité



In medical diagnosis, test sensitivity is the ability of a test to correctly identify those with the disease, whereas test specificity is the ability of the test to correctly identify those without the disease

https://kennis-research.shinyapps.io/Bayes-App/

Algorithme de détection de fruits (programmation python)

Vous l'aurez compris le cœur de l'algorithme va reposer sur

la comparaison des contenus de dfr.predicted et de dfr.fruits ou dfr.actual

dfr.fruits (ou dr.actual) : réalité/vérité

dfr.predicted : prédictions

dfr.predicted → dfr.actual ?

Matrice de confusion (tableau croisé)

```
# dépend de la bibliothèques "sklearn" et du module "metrics"
confusion_matrix = pd.crosstab(df3.actual, df3.predicted,_
                               rownames=['Actual'], colnames=['Predicted'])
print (confusion_matrix)
```

Performance de l'algorithme

(Matrice de confusion, accuracy, precision, recall, % d'erreurs; ...)

```
from sklearn import metrics
confusion_matrix = pd.crosstab(df3.actual, df3.predicted,_
                               rownames=['Réel'], colnames=['Prédit'])
print (confusion_matrix)
```

```
Prédit  autre  orange
Réel
autre      13       0
orange      0       3
```

Performance de l'algorithme

(Matrice de confusion, accuracy, precision, recall, % d'erreurs; ...)

```
# dépend de la bibliothèques "sklearn" et du module "metrics"
confusion_matrix = pd.crosstab(df3.actual, df3.predicted,_
                               rownames=['Actual'], colnames=['Predicted'])
print (confusion_matrix)
```

```
Predicted  autre  orange
Actual
autre        143       0
orange         8     120
```

```
report_pred=metrics.classification_report(df3.actual, df3.predicted)
print(report_pred)
print(metrics.accuracy_score(df3.actual, df3.predicted))
#taux d'erreur
print(1.0 - metrics.accuracy_score(df3.actual, df3.predicted))
```

```
              precision    recall  f1-score   support

       autre       0.95      1.00      0.97       143
      orange       1.00      0.94      0.97       128

    accuracy                           0.97       271
   macro avg       0.97      0.97      0.97       271
weighted avg       0.97      0.97      0.97       271

0.9704797047970479
0.029520295202952074
```

Algorithme de détection de fruits (programmation python)

Et pour les autres fruits ?
(le faire progressivement : ajouter pêches et raisin)

si…sinon si… sinon si…. sinon

if…elif…elif…else

```
if (dfr.fruit[i]=="orange") :  dfr.actual[i] = "orange"
elif (dfr.fruit[i]=="??") … ???
```

1. Rappels échantillon / population

2. Continuer l'apprentissage sous contrainte
   1 seule courte étape
   (on fait comprendre à la machine ce qu'elle doit reconnaître)

3. matrice de confusion
   "accuracy" (+) évaluer la performance de l'algo

4. À vous d'écrire la suite (2 fruits de plus reconnus)
   (⇒ Fichier (+) conséquent 270 valeurs)
   (+) évaluation performance algo

5. → correction proposée

6. → vers le Machine learning → explorer le contenu du Fichier.

7. jeu n°3

→ Rappel situationnel

→ Algorithme contraint

↳ aller plus vers "l'apprentissage" réflexion 1

→ réécrire le coeur de l'algo

→ évaluer la performance de la reconnaissance

Matrice de confusion & accuracy

→ vers le ML → réflexion 2

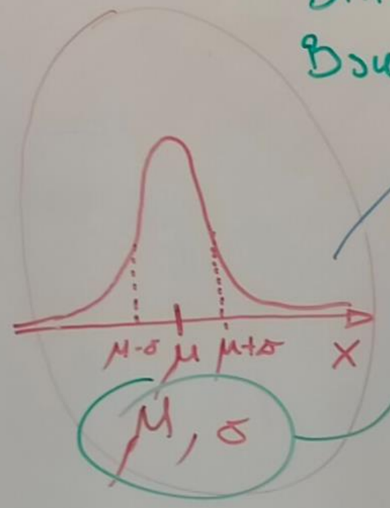
→ reconnaissance auto Fruits [orange, pêche, raisins]
(VOUS) + correction

→ ~~jeu~~ Bilan séance

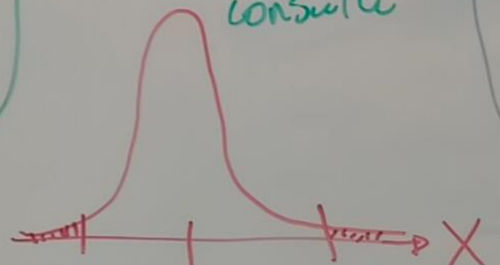
X : VA d'étude idéal $n \geq 2$

$B_{inf} = \mu - \varepsilon\sigma$

$B_{sup} = \mu + \varepsilon\sigma$



Fichier externe consulté



Population $\mu - \varepsilon\sigma$ μ $\mu + \varepsilon\sigma$

(+ caractérisée

↳ paramètre position ex μ

paramètre de dispersion : σ ou σ^2

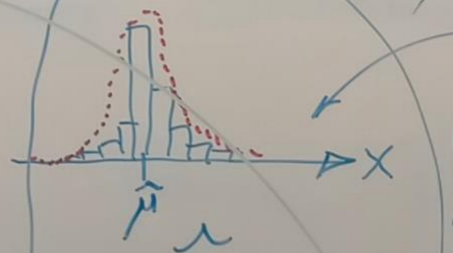
Facteur de forme

distribution (normale ?)

N inconnu

$\overline{X_n}$

réalisation en classe



$x_1, x_2, \ldots, x_k \ldots, x_n$

$\hat{\mu}$: estimation non biaisée (ponctuelle) de μ

$n \geq 30$

calculs (exacts)

$\hat{\mu}$

$\hat{\sigma}^*$

$\hat{\sigma}^{*2} = \frac{n}{n-1} S_x^2$ variance calculée de l'échantillon

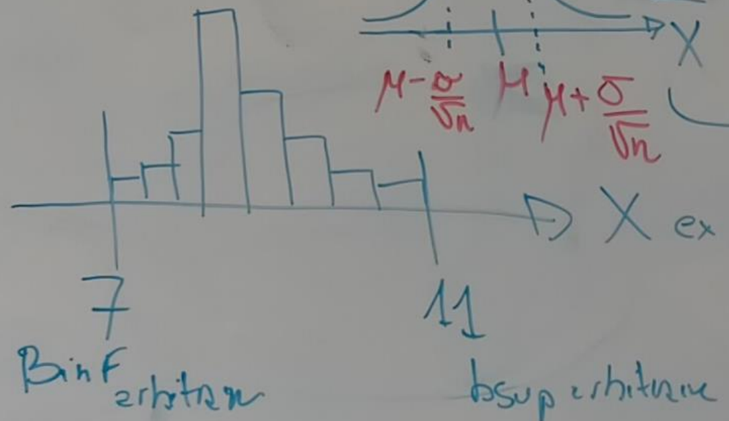
$F(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left[\frac{x-\mu}{\sigma}\right]^2}$

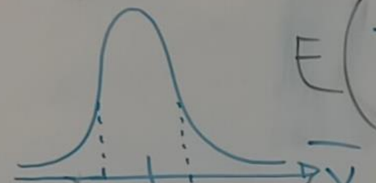
Cœur de l'algo

si $\hat{\mu} \in [B_{inf_A}, B_{sup_A}[\Rightarrow$ orange

estimateur $\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} x_i$

n échantillon de taille n



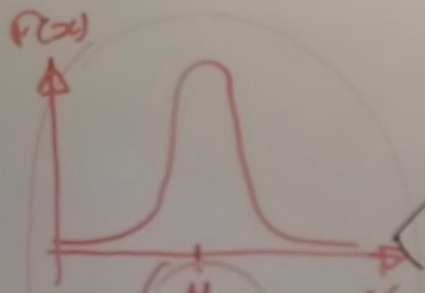


$E(\overline{X}) - \mu = 0$

$\overline{X}$ estimateur non biaisé de μ

↳ On peut construire un intervalle de confiance en utilisant $\overline{X}$

X : VA d'étude



Si connu

Population (caractère)

N inconnue (très grande) $X \rightsquigarrow N(\mu, \sigma)$

μ, σ inconnus

position : mode, min, max, moy

dispersion : σ, σ^2

Forme : Kurtosis, Skewness

inférence

prédiction

modélisation

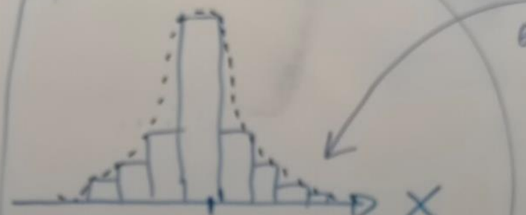
Généralisation

STATS

ML

Biais (récolte, stress hydrique, etc...)

redistribution en classes



$[x_1, x_2) \cdots x_k, x_n]$

calculs

$\hat{\mu}$ estimation ponctuelle de μ ($\hat{\mu} \simeq \mu$)

$\hat{\sigma}$ estimation ponctuelle de σ

correction de biais

$\hat{\sigma}^{*2} = \frac{n}{n-1} \hat{\sigma}^2$

$E(\bar{X}_n) - \mu = 0$

échantillon de taille n

$n \geq 30$

(objet de travail en ML)

$\bar{X}_n$ estimateur

$\bar{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i$

Coeur algo ML

7 < diam < 11 contrainte

min < diam < max

$F_2(\mu,\sigma)$ < diam < $F_1(\mu,\sigma)$

recherché recherché

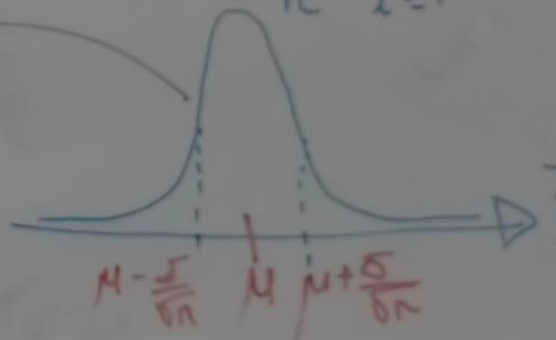
$f_1(?) < \text{diam} < f_2(?)$

IC encadrement de μ

IC_α

$\mu \in [B_{inf}, B_{sup}]$

IC

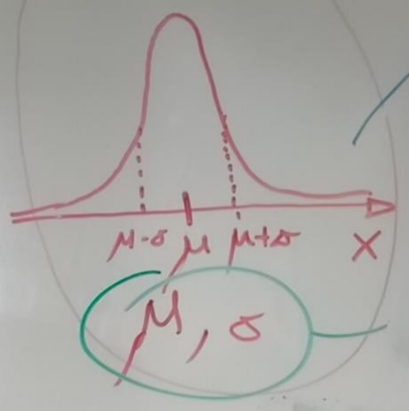


$F(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{1}{2}\left[\frac{x-\mu}{\sigma}\right]^2}$

X : VA d'étude idéale

$Binf = \mu - \varepsilon\sigma$

$Bsup = \mu + \varepsilon\sigma$



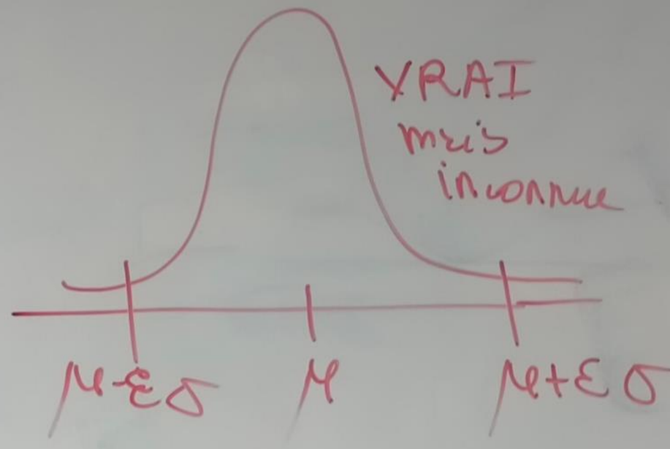
Machine learning

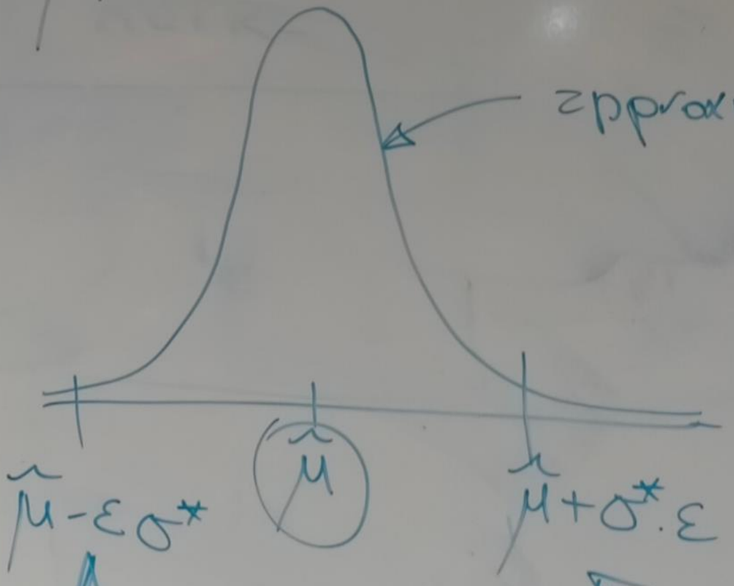
Blue = Bizi

travail sur échantillon

$\mu \longrightarrow \hat{\mu}$

la machine lit le Fichier elle récupère $\hat{\mu}$ et $\hat{\sigma}^*$

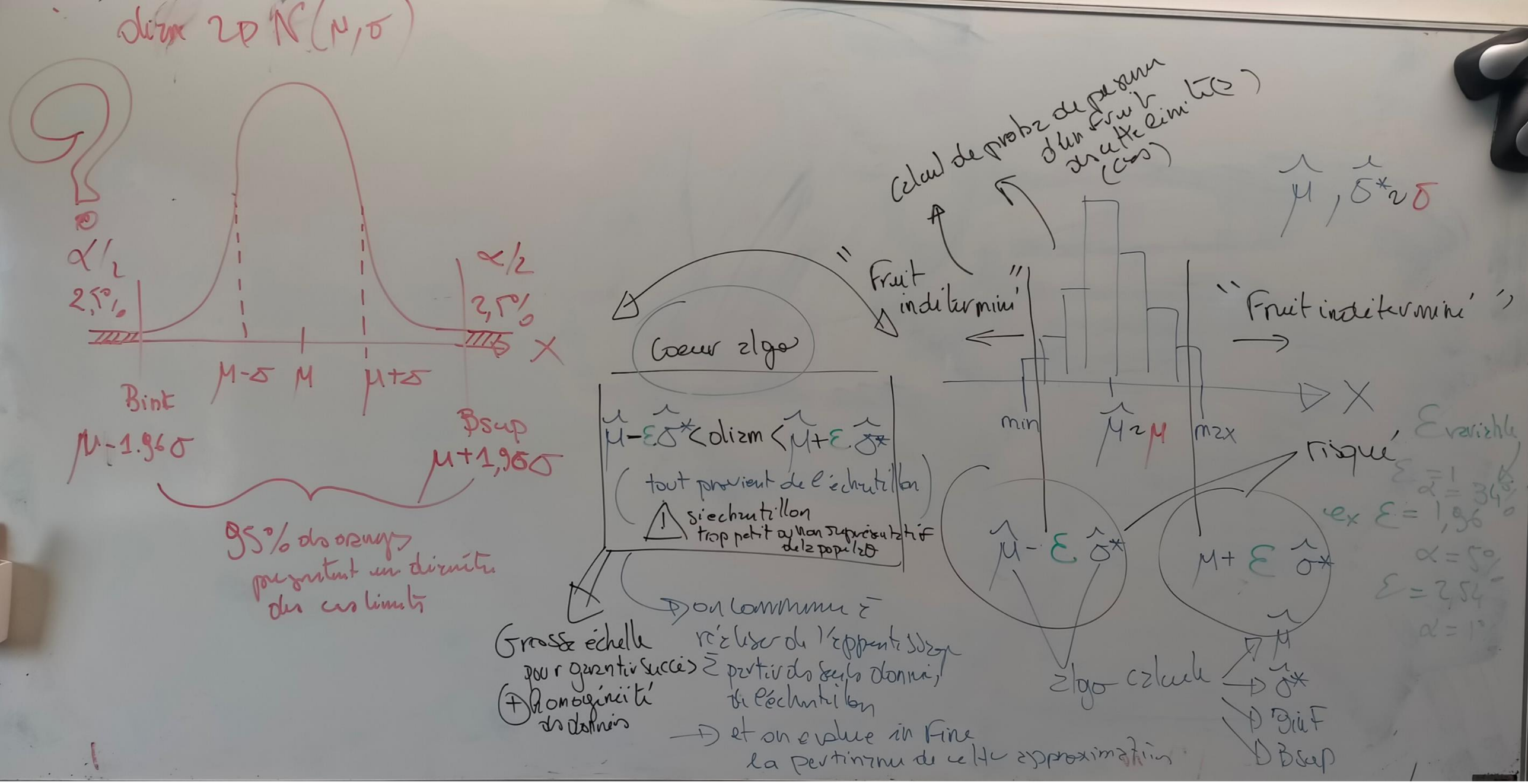




Binf

Bsup

CL dizm $\in \left[\hat{\mu}-\varepsilon\hat{\sigma}^*, \hat{\mu}+\varepsilon\hat{\sigma}^*\right]$


dizm 2D N(μ,σ)
α/2
2,5%
α/2
2,5%
X
μ-σ
μ
μ+σ
Binf
μ-1,96σ
Bsup
μ+1,96σ
95% des cas qui permettent un diagnostic des cas limites
Coeur algo
Fruit indéterminé
Calcul de probas de passer d'un événement aléatoire (cas)
μ̂, δ̂*2δ
Fruit indéterminé
μ̂ - ε δ̂* < dizm < μ̂ + ε δ̂*
tout provient de l'échantillon
si échantillon trop petit qu'il n'est plus représentatif de la population
Grosse échelle pour garantir succès + homogénéité des données
On commence à réduire la taille de l'échantillon à partir des seuls données de l'échantillon
et on évalue in fine la pertinence de cette approximation
min
μ̂ = μ
max
risqué
ε variable
ex ε = 1,96
α = 5%
ε = 2,54
α = 1%
μ̂ - ε δ̂*
μ̂ + ε δ̂*
algo calcule
μ̂
δ̂*
Binf
Bsup

Matrice de confusion

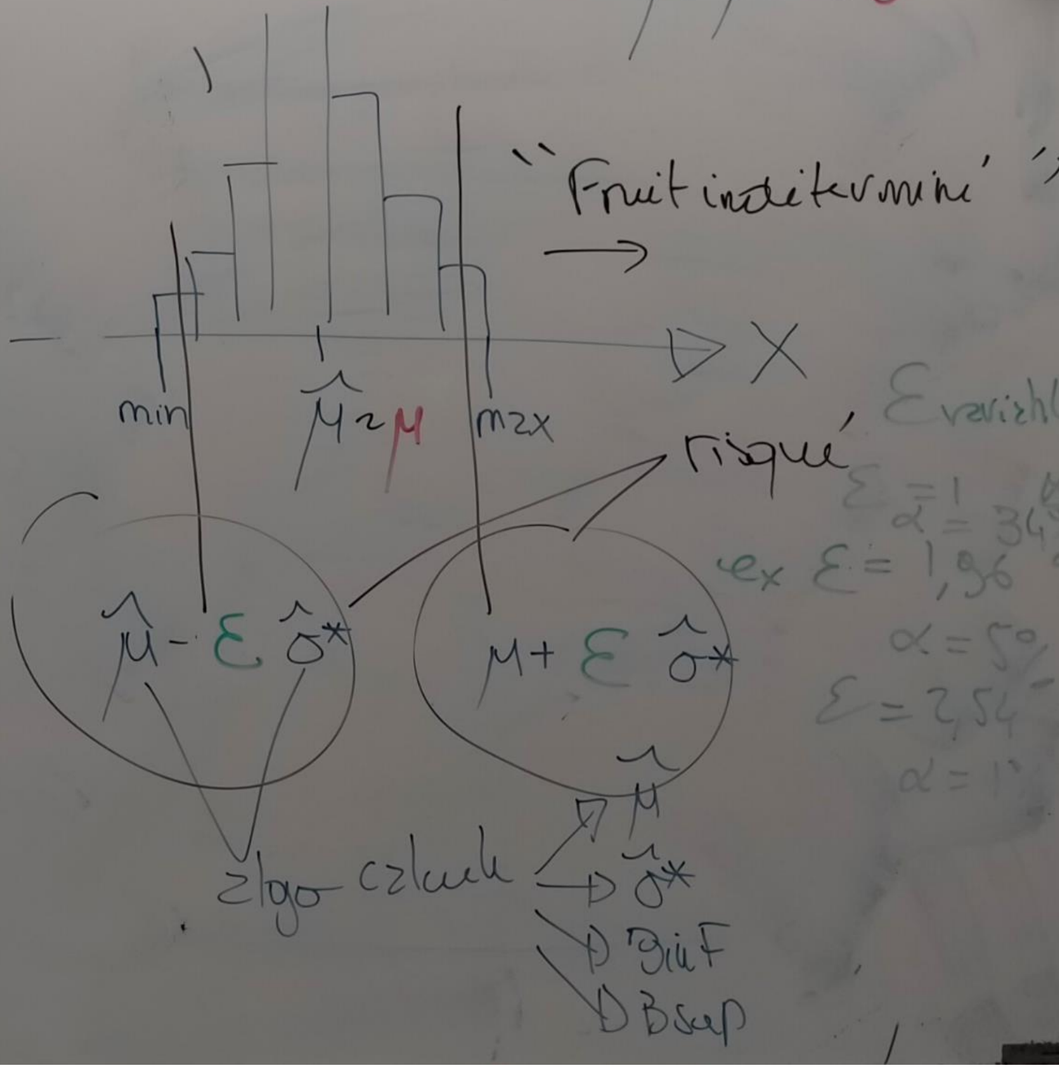
$$\text{accuracy} = \frac{VP + VN}{VP + VN + FP + FN}$$

$\hat{\mu}, \hat{\sigma}^* \sim \delta$

tx d'erreur = 1 - accuracy

PREDIT	Réel : ORANGE	Réel : AUTRE
ORANGE	VP	FP
AUTRE	FN	VN

→ Performance algo calculable



ε variable

$\varepsilon = 1$ $\alpha = 34\%$

ex $\varepsilon = 1,96$ $\alpha = 5\%$

$\varepsilon = 2,54$ $\alpha = 1\%$