

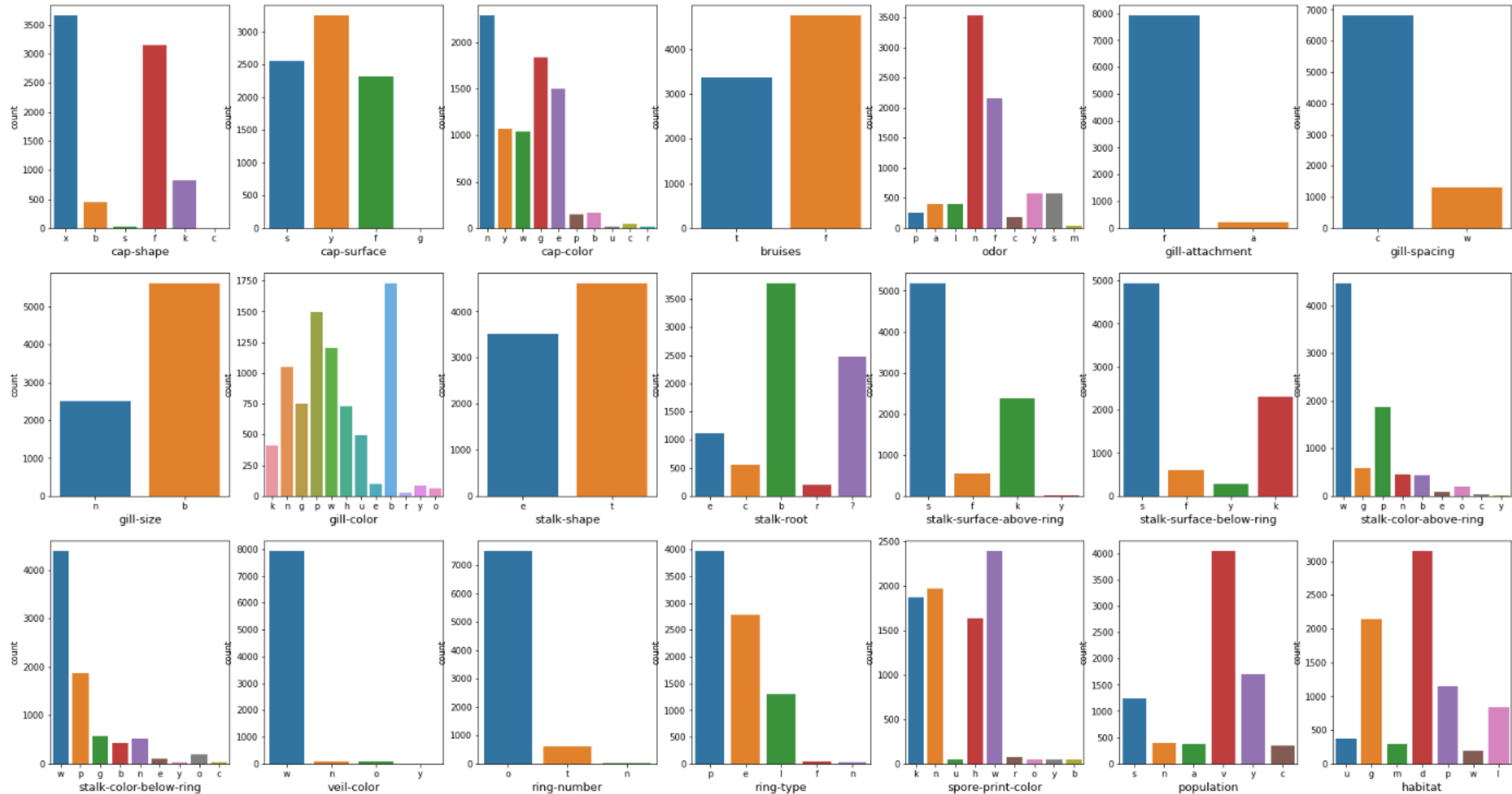
# Cas Kaggle

# Mushrooms classification

Albert Roca i Llevadot

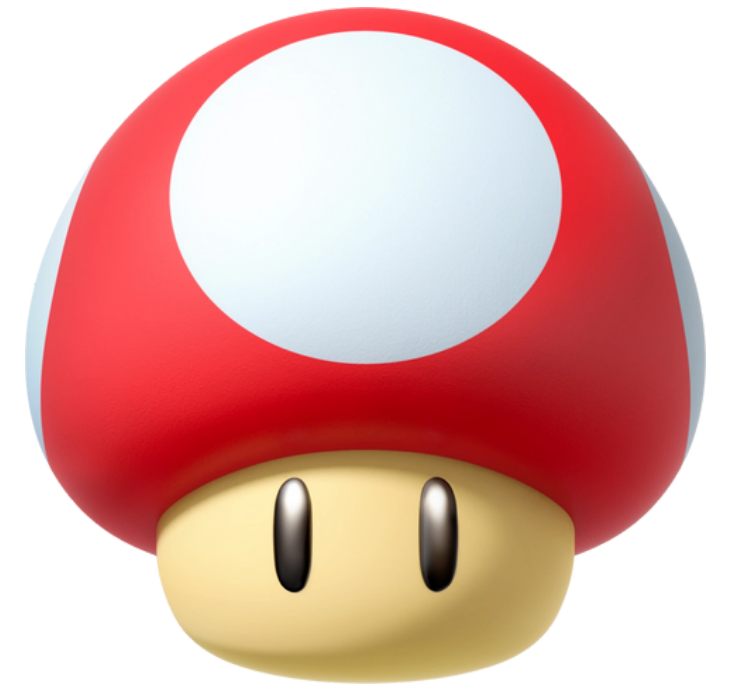
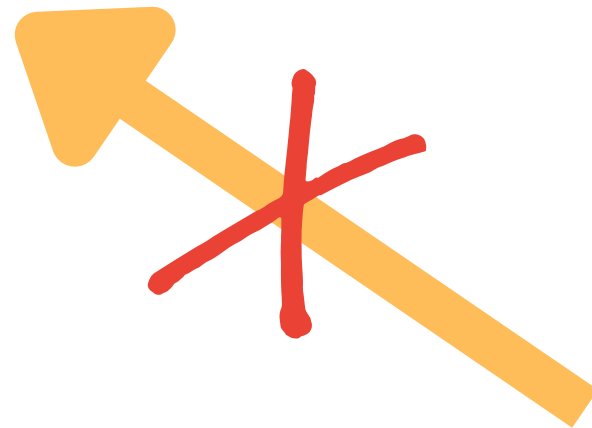


# Un dataset amb variables categòriques

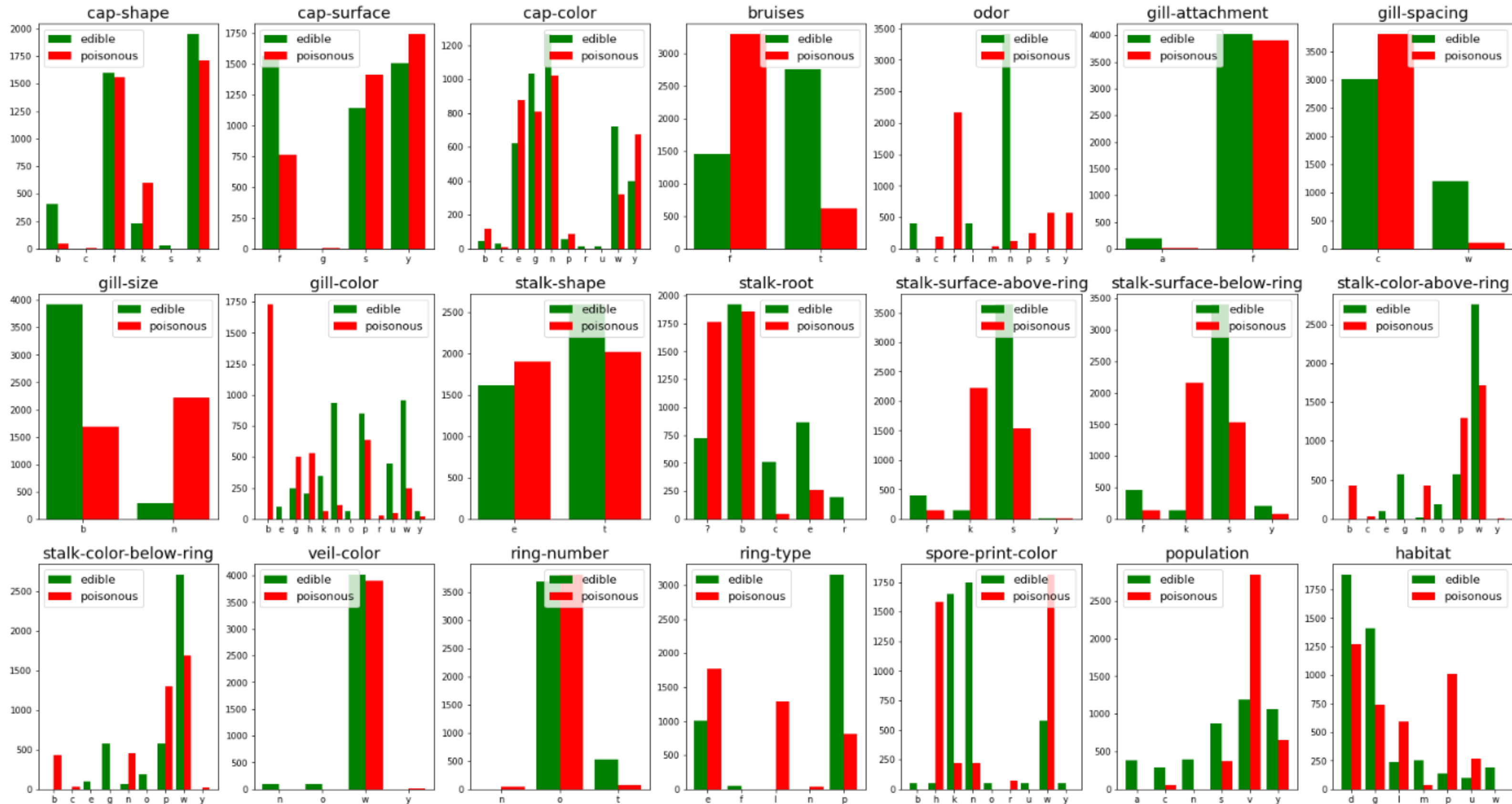


# Objectiu:

## Classificar els bolets entre verinosos i comestibles

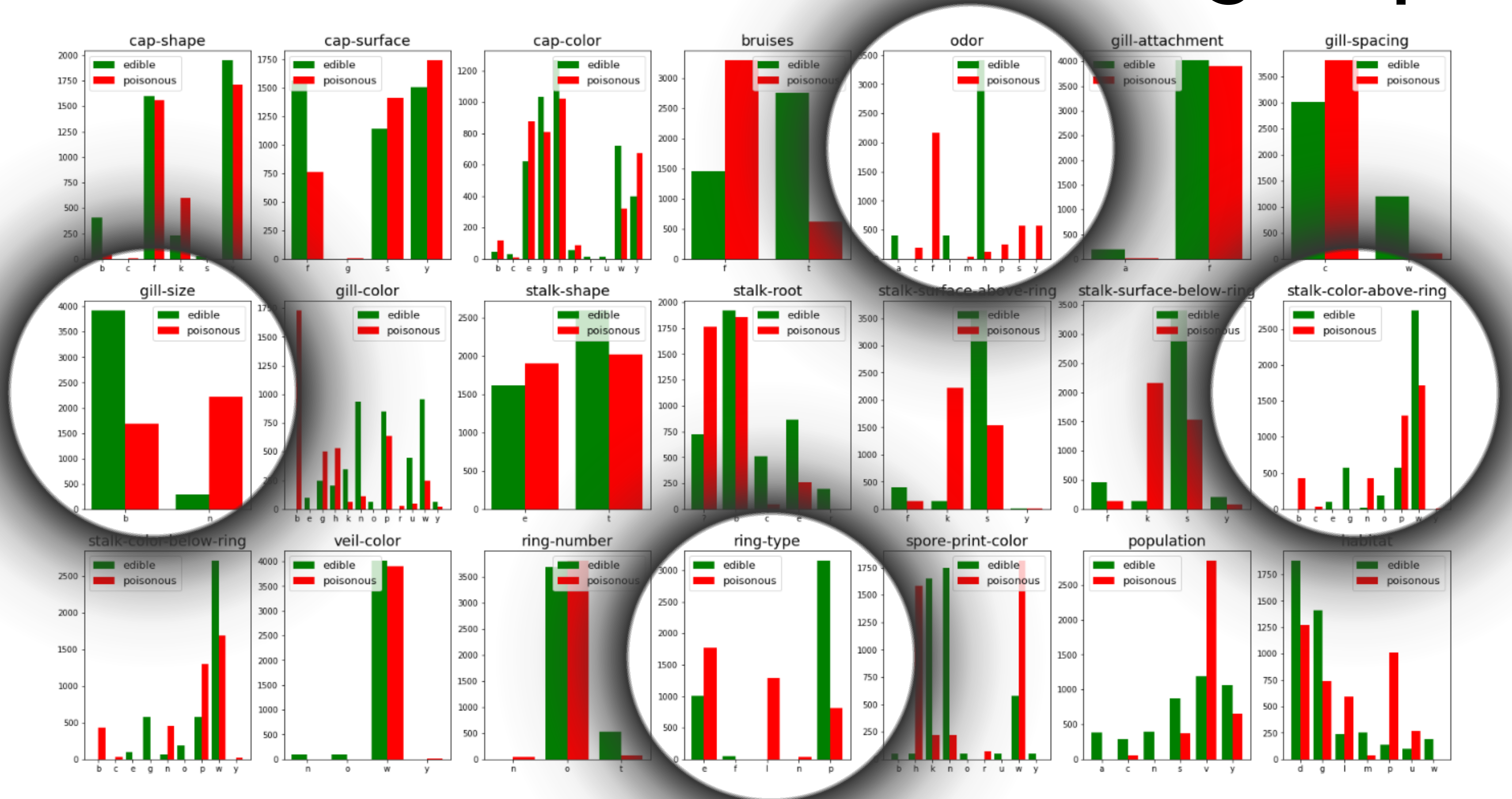


# Un dataset amb variables categoriques

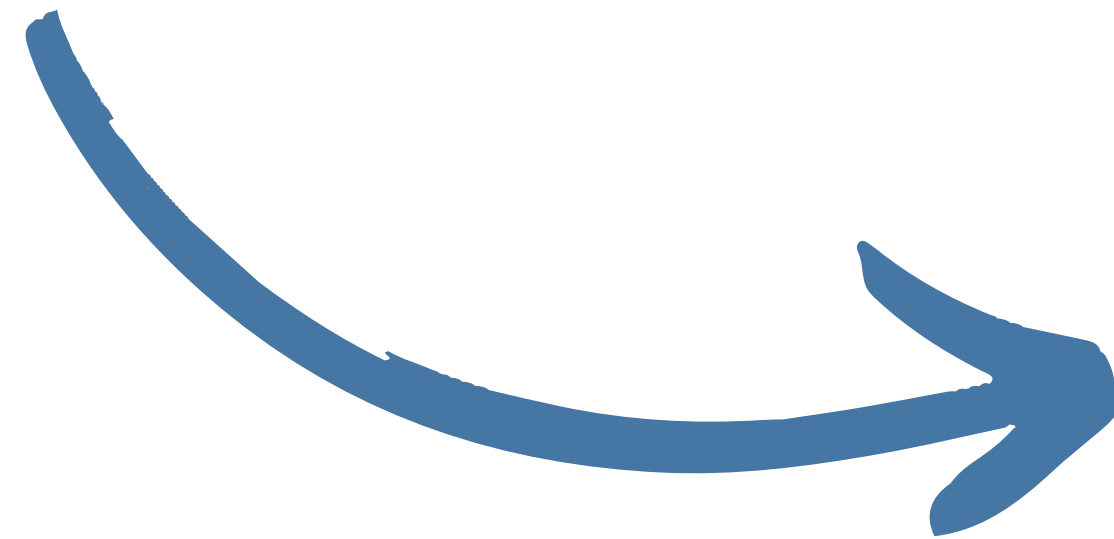




# Un dataset amb variables categoriques



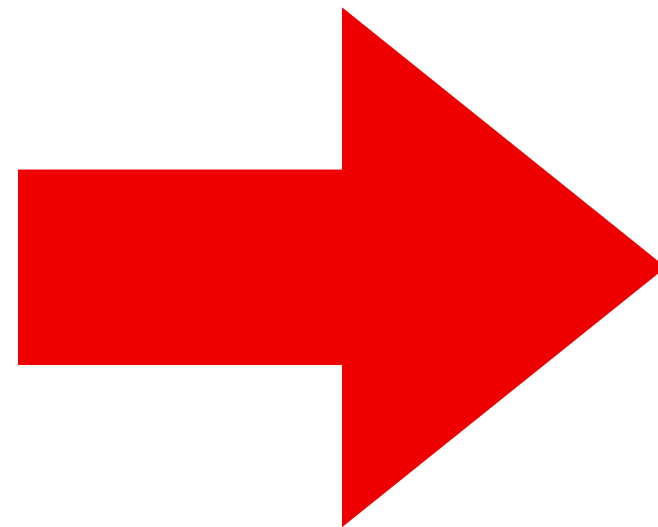
# Variables catégoriques



## Dummies

**21**

**variables**



**116**

**variables**

# Primera classificació

**ACCURACY**

*100%*

Amb tots els models escollits

Malgrat l'overfitting, l'accuracy és impecable.

**Veig que és una variable molt fàcil de predir amb  
les dades que tinc**

# Aleshores, que faig ?

Decideixo basar el treball en :

1

Quins atributs van millor i com augmenta l'accuracy amb cada un d'ells

2

Reduir la dimensió amb t-SNE i PCA.

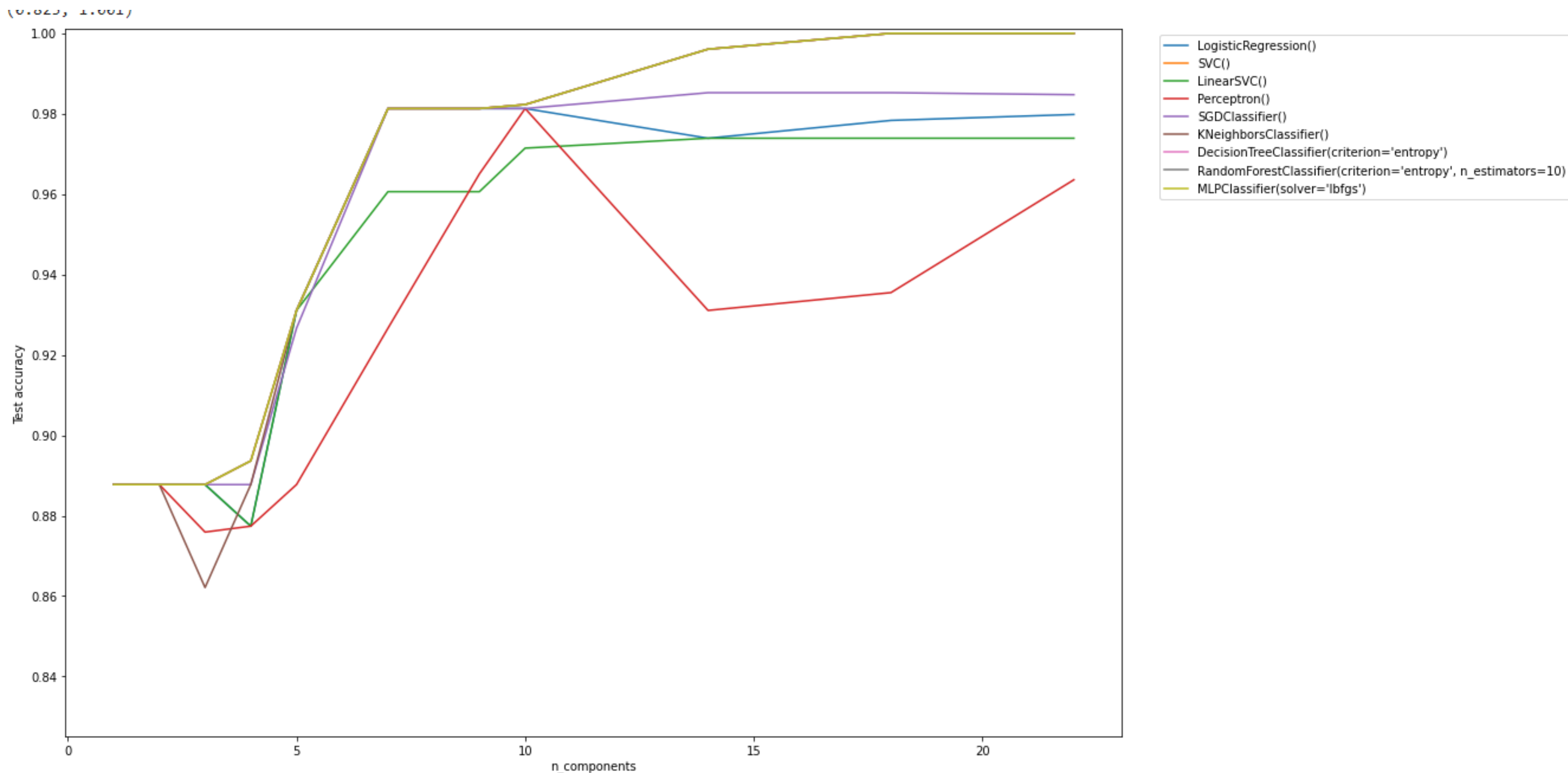
3

Visualitzar els resultat i veure fins a on puc arribar mantenint aquesta alta accuracy.



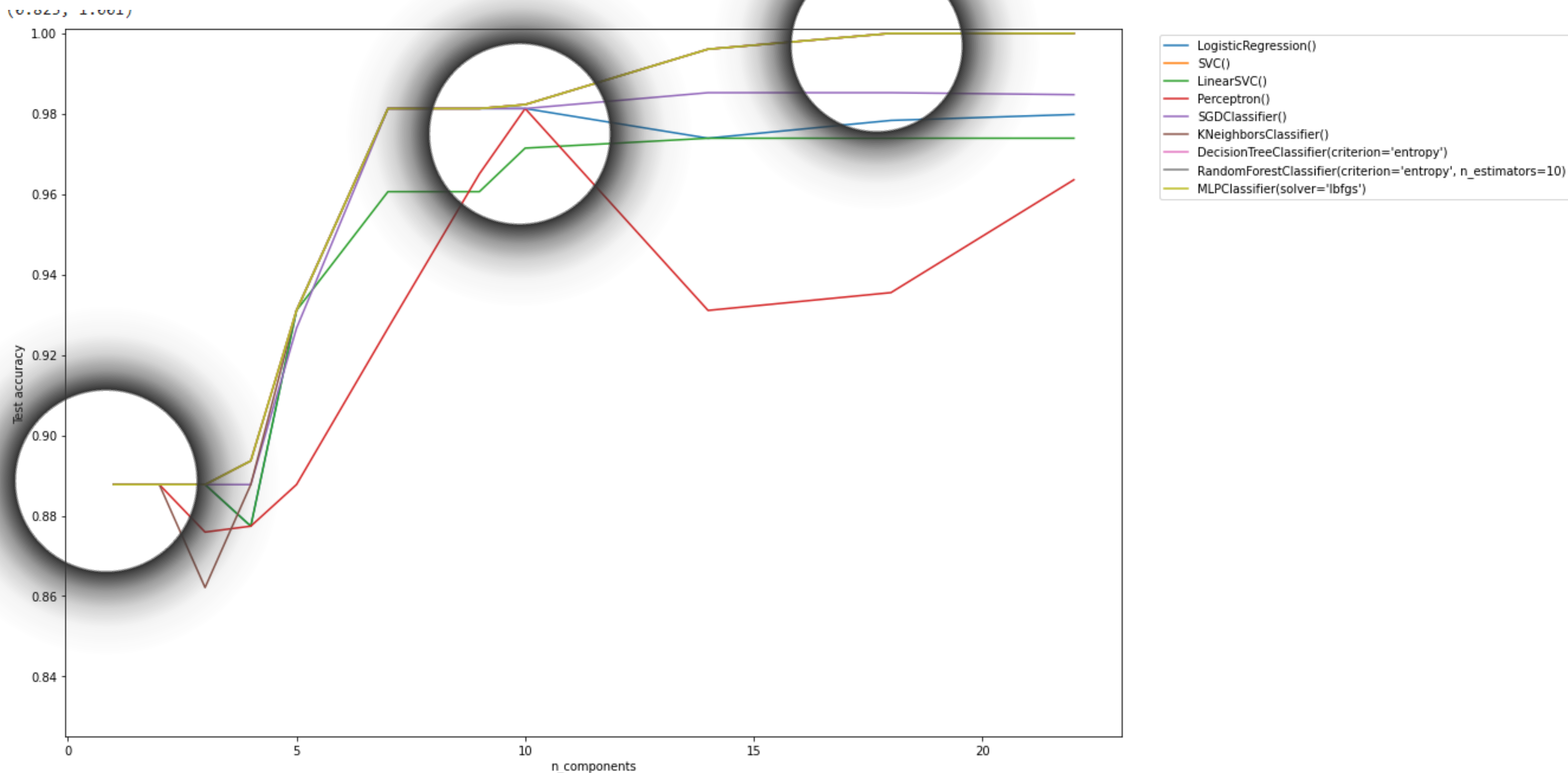
1

# Quins atributs van millor i com augmenta l'accuracy amb cada un d'ells



1

# Quins atributs van millor i com augmenta l'accuracy amb cada un d'ells



# Millors atributs en solitari

Pel que fa la practica sembla el més interessant

Nom	Accuracy	Falsos Comestibles	Signe de la correlació
odor_n	0.88774	0.00410	1
odor_f	0.78877	0.07041	-1
stalk-surface-above-ring_k	0.78385	0.06663	-1
stalk-surface-below-ring_k	0.76809	0.07139	-1
ring-type_p	0.76711	0.03299	1
gill-size_b	0.75726	0.06910	1
gill-size_n	0.75726	0.06910	-1
bruises_f	0.75283	0.02527	-1
bruises_t	0.75283	0.02527	1
stalk-surface-above-ring_s	0.75234	0.06089	1

# Millors atributs en solitari

L'atribut millor i amb menys falsos negatius  
és aquest:

**odor = none**

**Els bolets que no fan olor, tels pots menjar  
sense preocupar-te massa**

# Resultats amb 6 atributs dels 116

**ACCURACY**

*97.5%*

Amb gairebé tots els models  
escollits

**Necessito reduir el dataset**

# PCA per reduir el dataset

Mantenint un  
90% de variança



**31** atributs

**ACCURACY**

*95% ~ 100%*

Alguns models empitjoren lleugerament  
Però la majoria segueix al 100%

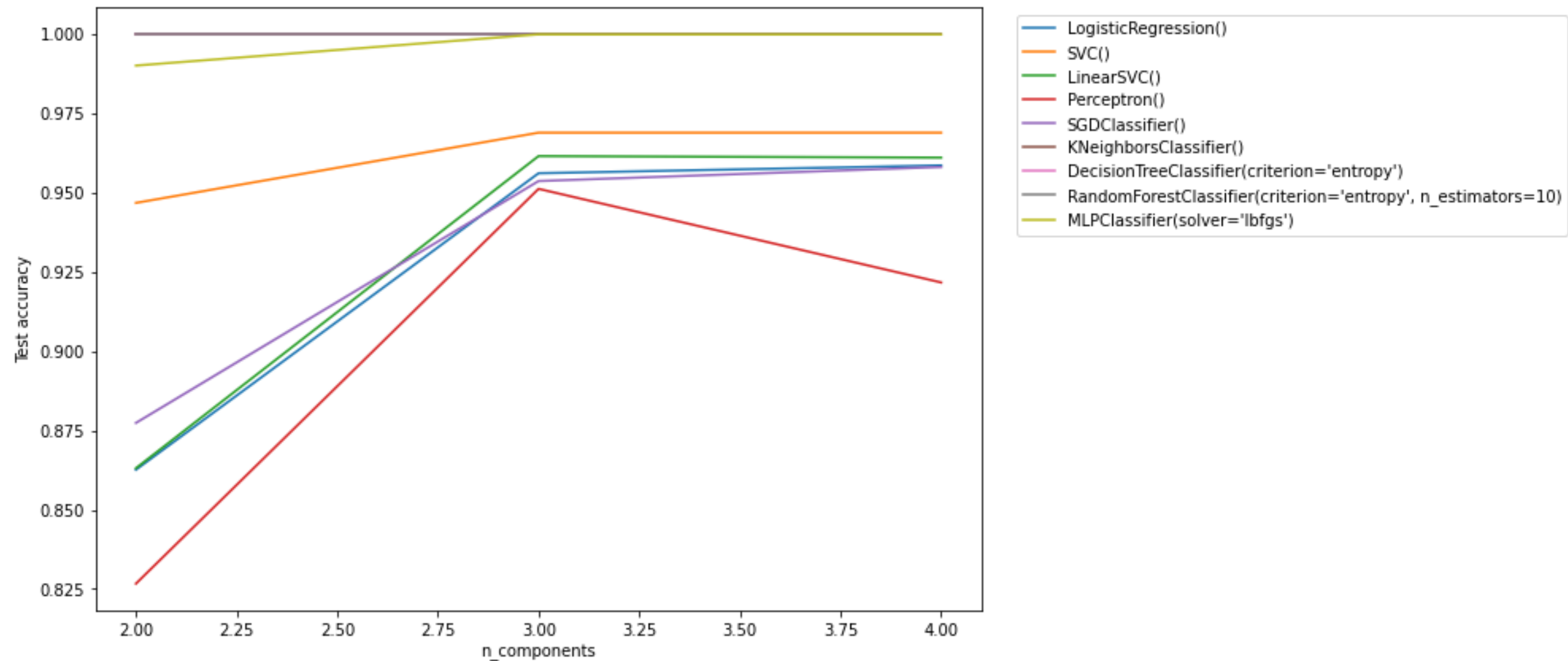


# 2

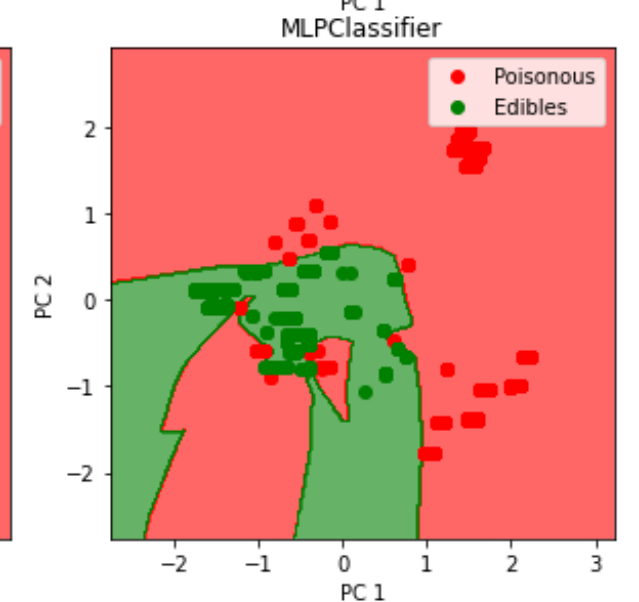
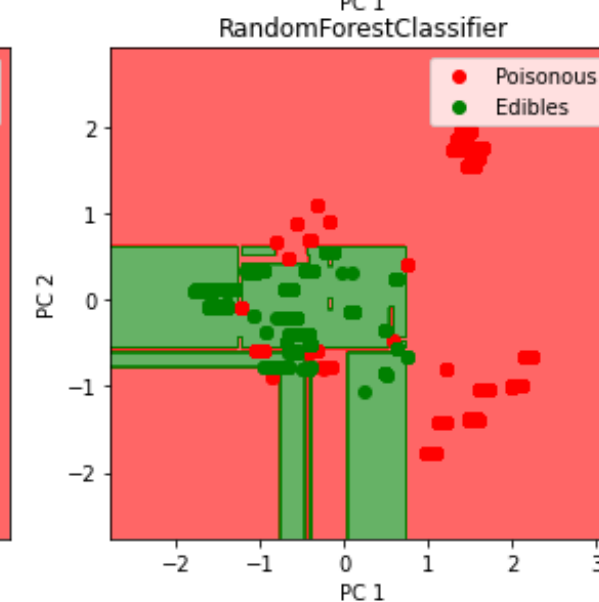
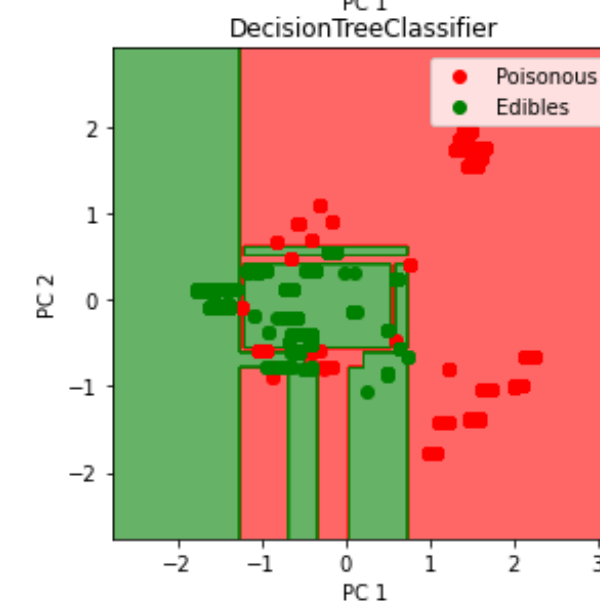
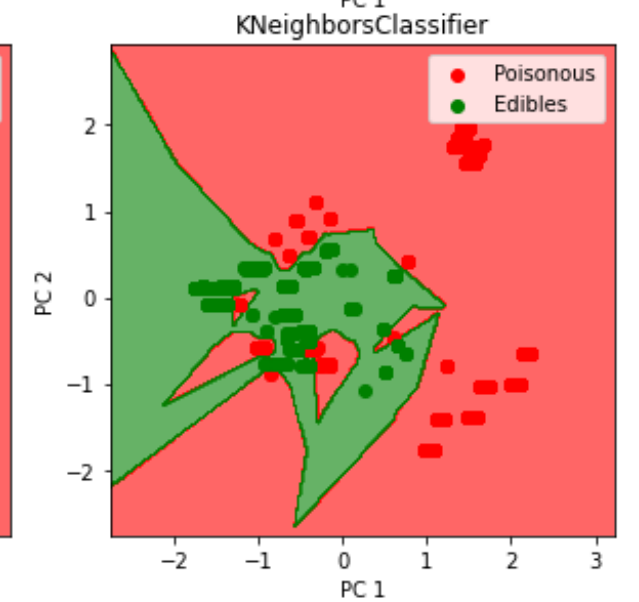
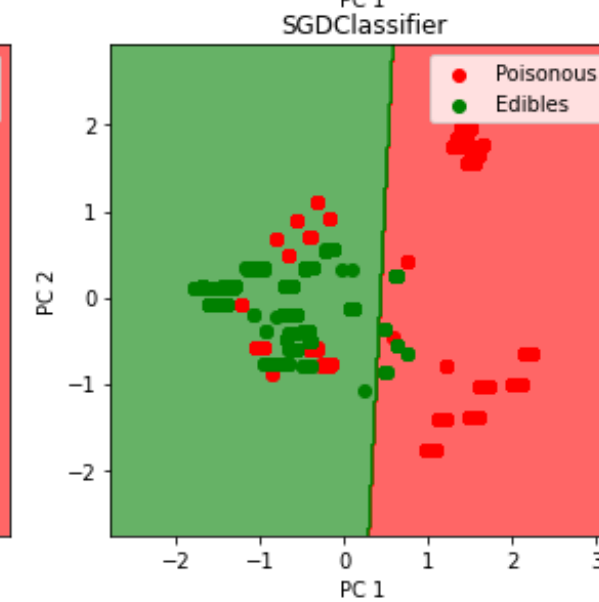
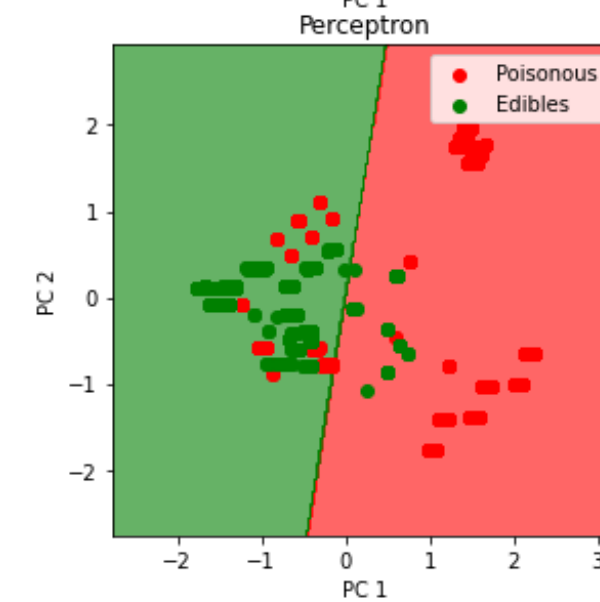
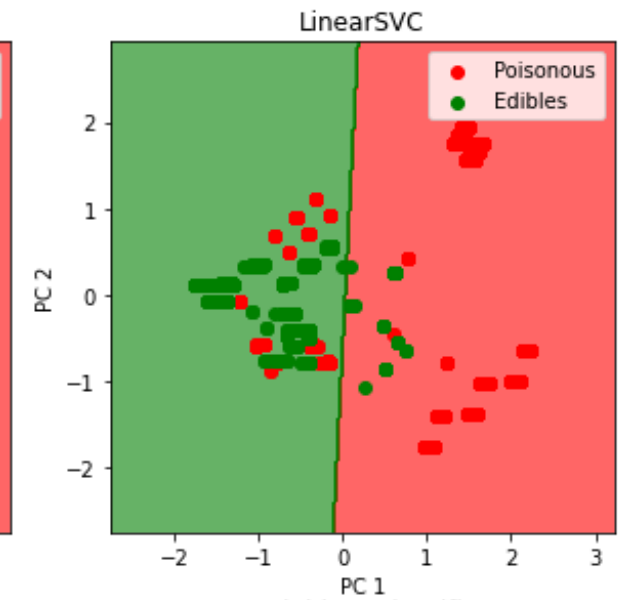
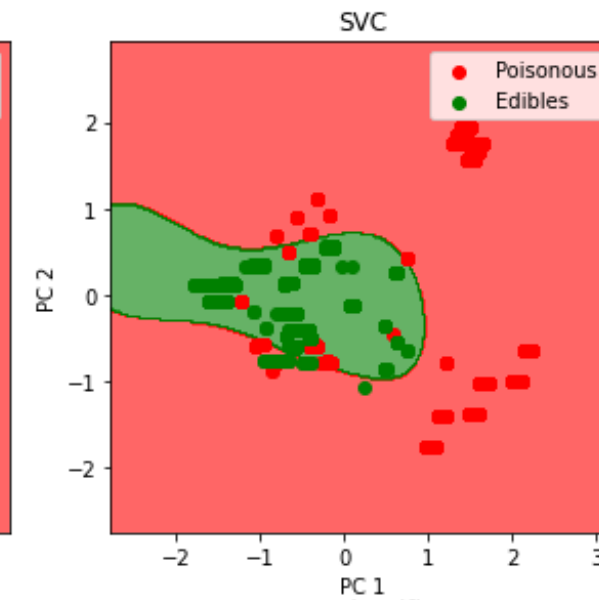
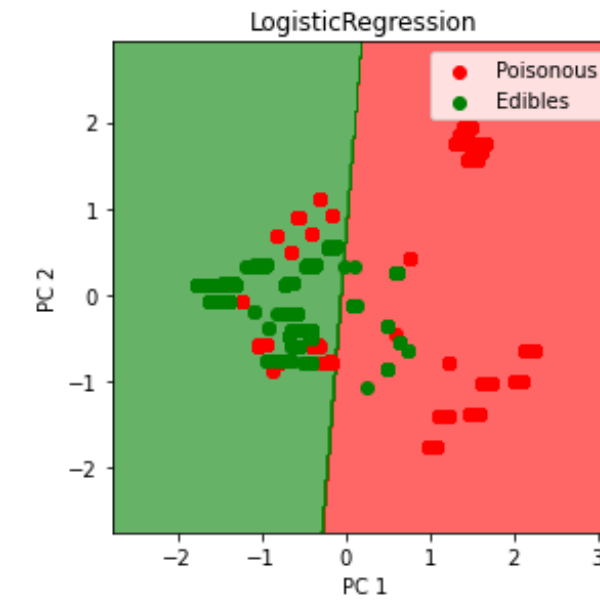
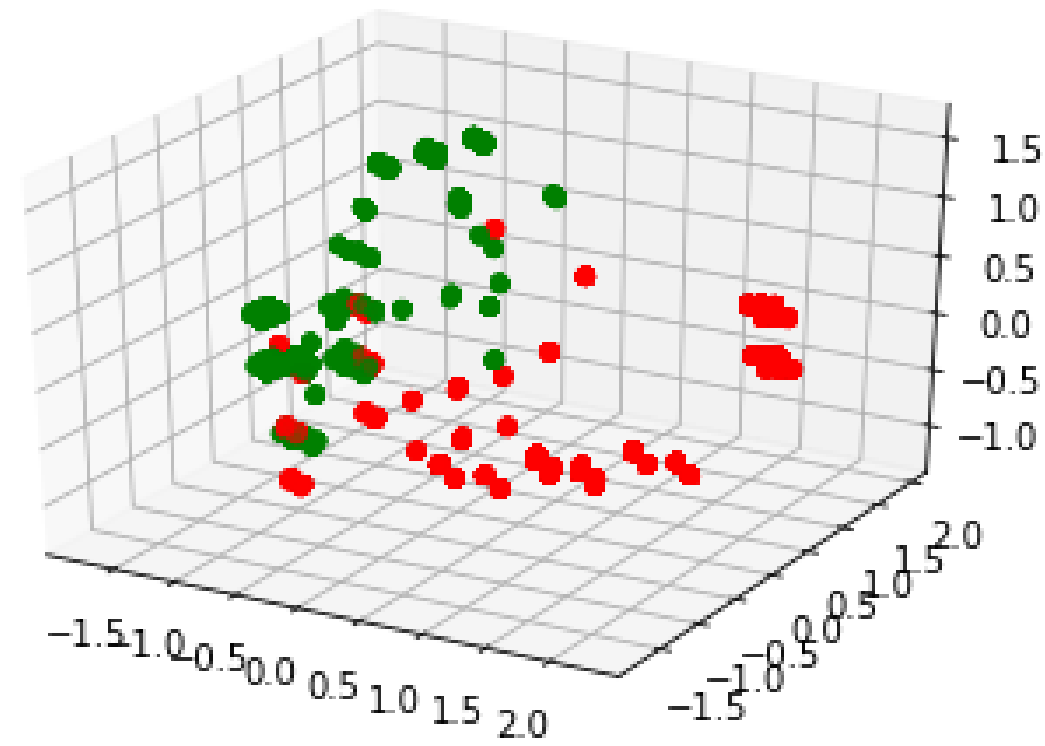
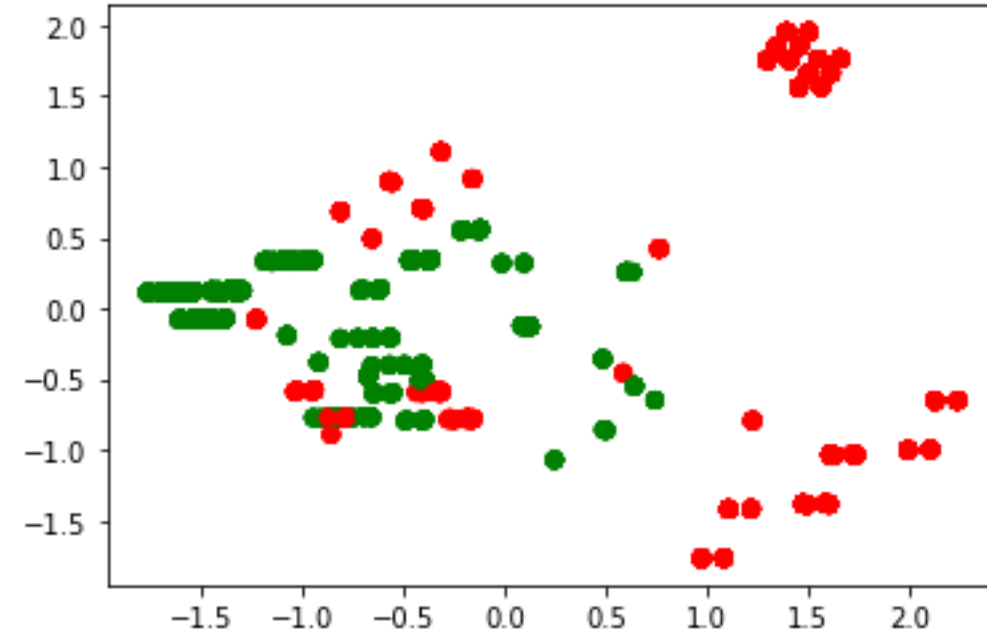
## PCA en dimensions 2 i 3

LogisticRegression	[0.86263 0.95618 0.95864]
SVC	[0.94682 0.96898 0.96898]
LinearSVC	[0.86312 0.9616 0.9611 ]
Perceptron	[0.82669 0.95126 0.92171]
SGDClassifier	[0.8774 0.95372 0.95815]
KNeighborsClassifier	[1. 1. 1.]
DecisionTreeClassifier	[1. 1. 1.]
RandomForestClassifier	[1. 1. 1.]
MLPClassifier	[0.99015 1. 1. ]

**ACCURACY**  
*100%*



# Com ho fa ?



# t-SNE en dimensions 2 i 3

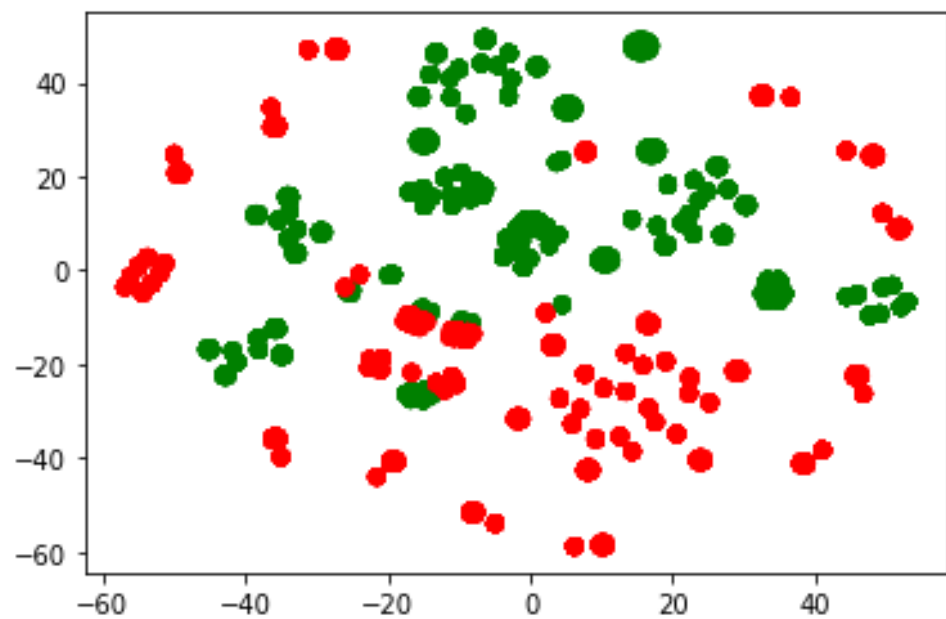
## **Alerta !**

L'algoritme t-SNE transforma tot l'espai de cerca de manera no lineal i irreversible.

Així que no és d'aprenentatge supervisat. Es pot avaluar l'accuracy però serà molt sensible a noves mostres (s'hauria de repetir l'algoritme cada cop).

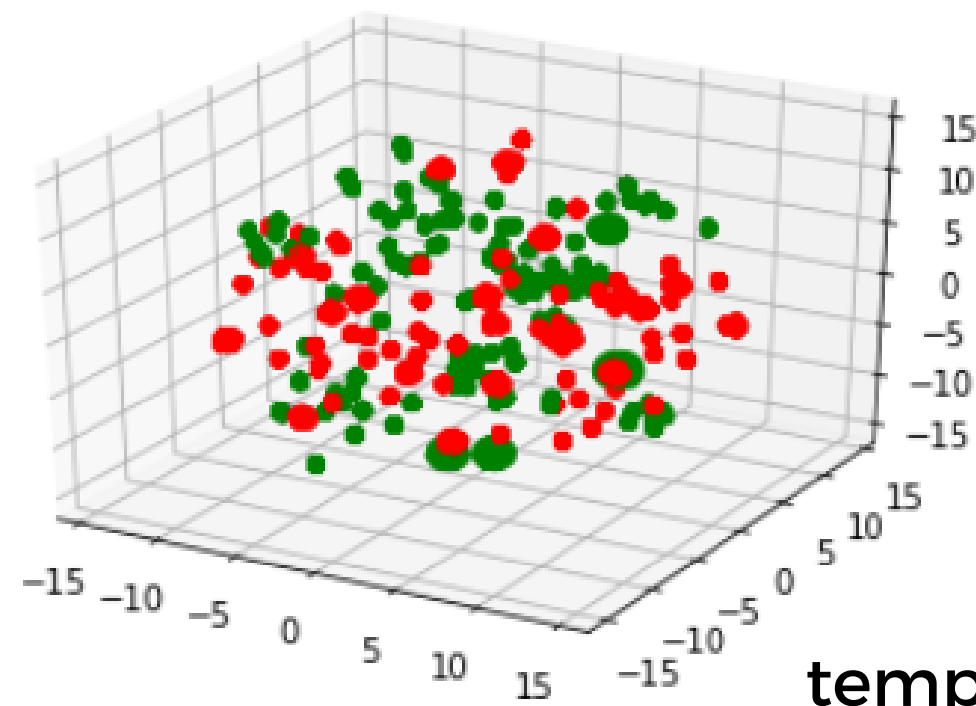
**És molt més lent**

TSNE(n\_components=2, n\_iter=750, perplexity = 100)

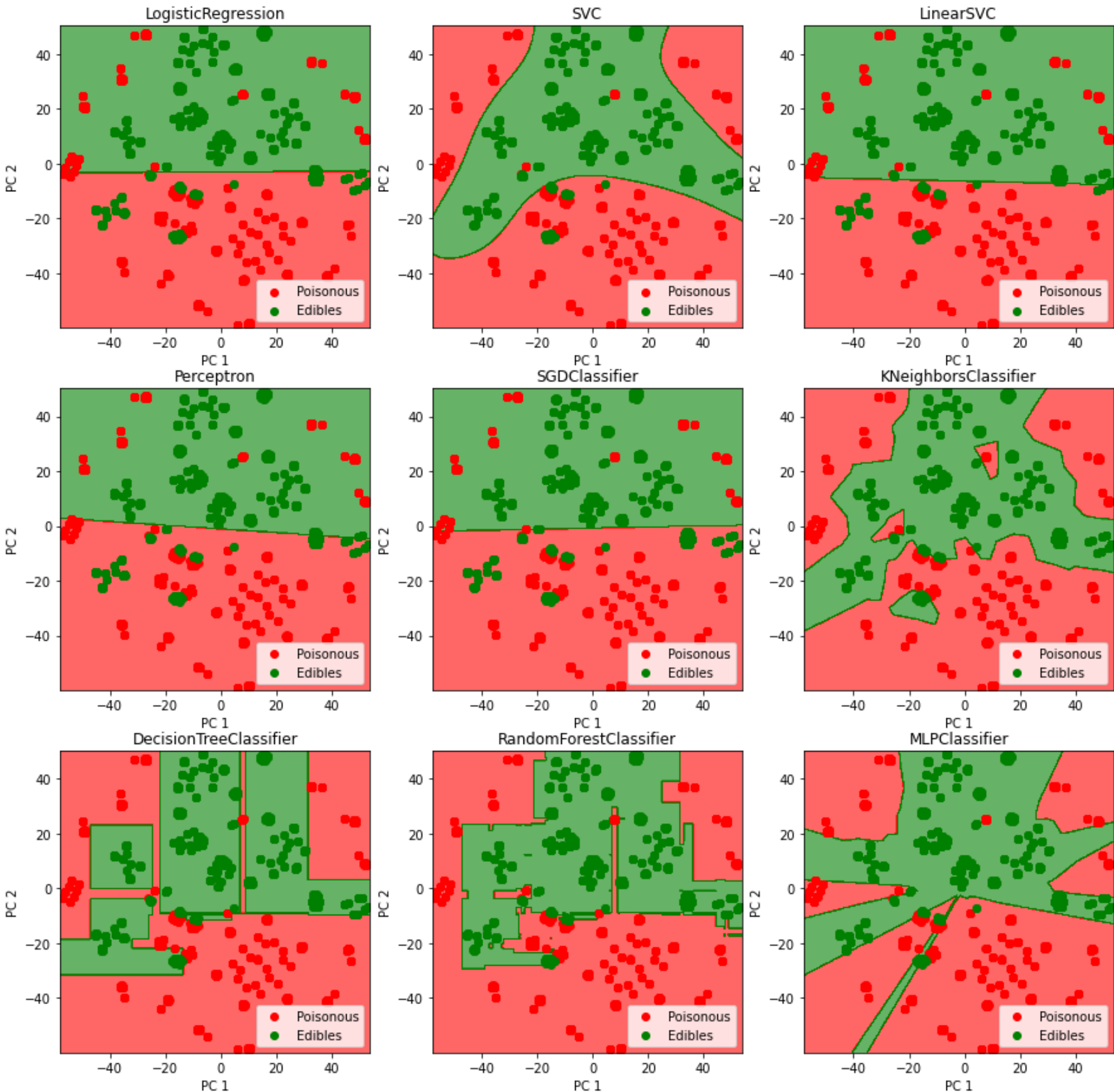


temps = 76 s

TSNE(n\_components=3, n\_iter=425, perplexity = 75)

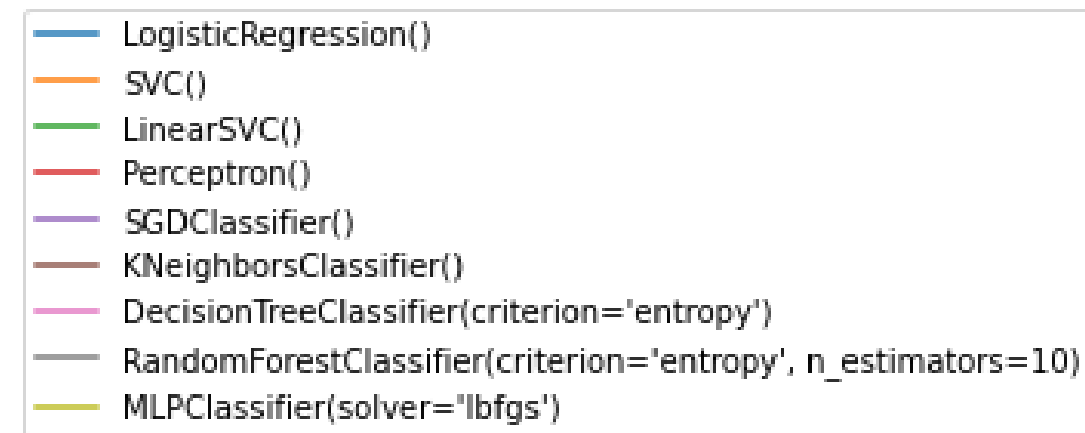
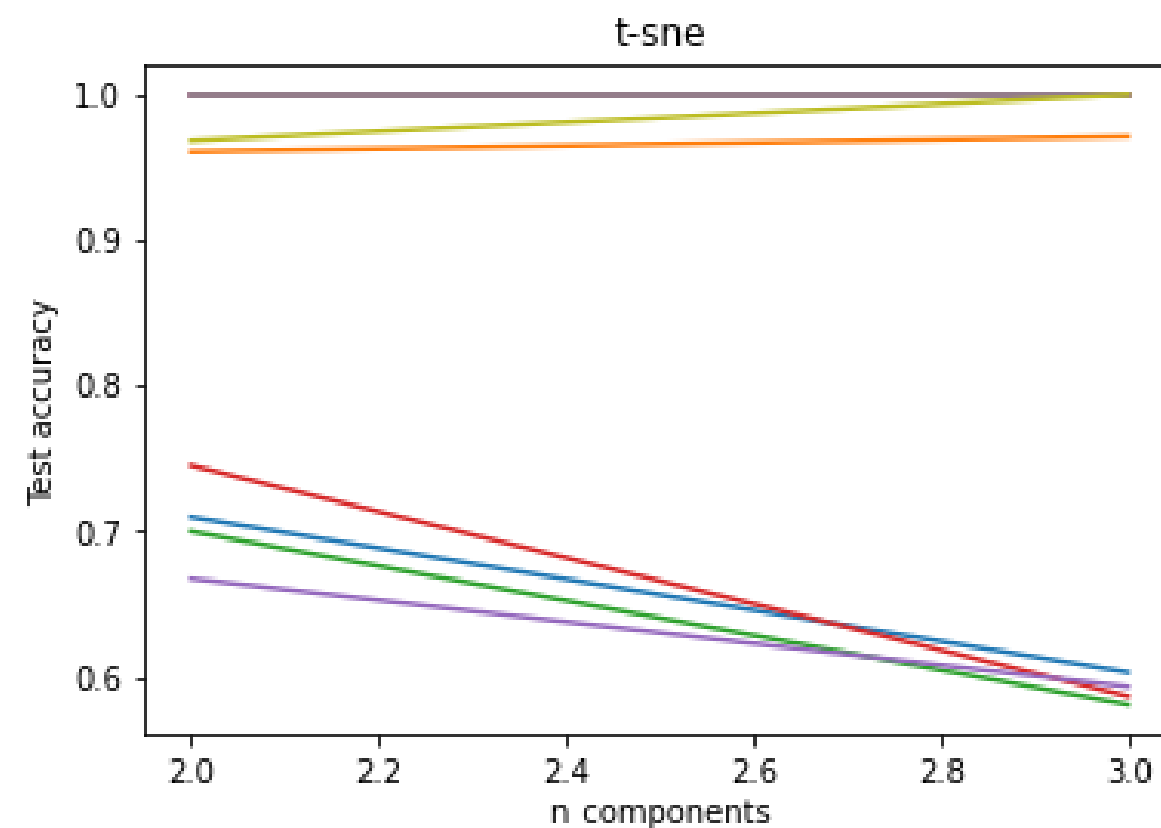


temps = 89 s



# t-SNE mesurar accuracy

Separo al nou espai en X\_train i X\_test.



```
[[0.71 0.60336]
 [0.9611 0.97129]
 [0.70015 0.58121]
 [0.74545 0.58696]
 [0.66765 0.59352]
 [1. 1. ]
 [1. 1. ]
 [1. 1. ]
 [0.96849 1. ]]
```

**ACCURACY**  
*100%*  
**EN ELS 4 MILLORS  
MODELS**

**PERO**  
per els models  
més senzills o  
lineals va pitjor

3

# Conclusions

**NO hi ha taula de resultats**

La gran majoria eren 100%

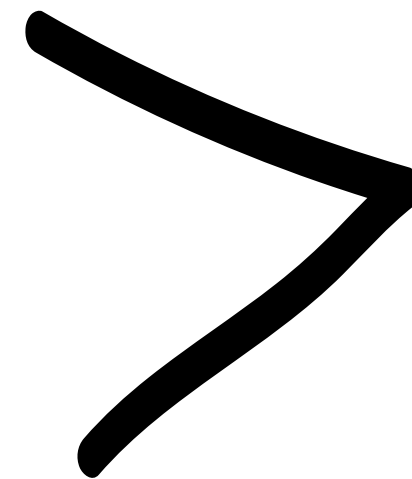
**Treballar amb un objectiu diferent**

Visualització de les dades

Modificar l'espai de cerca

Importancia dels arguments

Graficar els models i transformacions



**resultats**