1. **Dividir en X, Y y hacer train 80% test 20%**

Partiendo de un dataframe con df donde tenemos los datos numéricos:

X = df.drop(“target”)

Y = df[“target”]

Y dividimos entre el 80% para entrenar y el 20% de validar el aprendizaje.

X\_train,X\_test,Y\_train,Y\_test= train\_test\_split(X,Y,train\_size=0.8)

El objetivo será entrenar el algoritmo con X\_train e Y\_train. Predecir X\_test y comparar la predicción Y\_pred vs Y\_test.

1. **Inicializa el modelo predicitivo**

nombre\_variable = Modelo()

1. **Búsqueda de los mejores hiperparametros: GridSearchCV()**

nombre\_variable2 = GridSearchCV(nombre\_variable,parameters)

Siendo parameters una variable de tipo diccionario donde contiene los hiperparametros a probar del modelo predictivo a usar.

1. **Entrenar/Ajustar el modelo**

nombre\_variable2.fit(X\_train,Y\_train)

1. **Predecir valor futuro**

y\_pred = nombre\_variable2.predict(X\_test)

1. **Comparar el valor real contra el valor predicho**

plot\_confusion\_matrix(Y\_test,y\_pred)

# Ejemplo con Regresión Logistica:

Teniendo previamente instalada la librería que desees usar e importado el modelo:

from sklearn.linear\_model import LogisticRegression

1. **Inicializa el modelo predicitivo**

lr = LogisticRegression()

1. **Búsqueda de los mejores hiperparametros: GridSearchCV()**

parameters ={'C':[0.01,0.1,1],

'penalty':['l2'],

'solver':['lbfgs']}

Logreg\_cv = GridSearchCV(lr,parameters)

Siendo parameters una variable de tipo diccionario donde contiene los hiperparametros a probar del modelo predictivo a usar.

1. **Entrenar/Ajustar el modelo**

Logreg\_cv.fit(X\_train,Y\_train)

1. **Predecir valor futuro**

y\_pred = Logreg\_cv.predict(X\_test)

1. **Comparar el valor real contra el valor predicho**

plot\_confusion\_matrix(Y\_test,y\_pred)