Exercice 2

auteurs : Costa Pedro, Delabays Louis, Guerne Jonathan

```
import numpy as np
```

```
# had to remove the ; at the end of every line to make it work
data_a = np.loadtxt('ex2-system-a.csv', delimiter=';')

dta_prior = data_a[:,:-1]
dta_y = data_a[:,-1].astype(int)
nb_class_a = len(np.unique(dta_y))
```

Bayes pour la classification

```
def get_preds(data_prior):
    return np.array([np.argmax(data_prior[i,:]) for i in range(len(data_prior))])

preds_a = get_preds(dta_prior)
```

Calcul de overall error rate

```
def ovrl_error(data_y,preds):
    ovrl_acc = len(data_y[data_y == preds])/len(data_y)
    return 1-ovrl_acc

er_a = ovrl_error(dta_y,preds_a)
print("Overall error rate : {}".format(er_a))
```

```
Overall error rate : 0.10729999999999995
```

calcul de la matrice de confusion

```
def get_confusion_matrix(data_y, preds,nb_class):
    confusion_matrix = np.ndarray((nb_class,nb_class))

    for i in np.unique(data_y):
        row = np.bincount(preds[data_y == i], minlength=nb_class)
        confusion_matrix[i] = row
    return confusion_matrix

cma = get_confusion_matrix(dta_y,preds_a,nb_class_a)
print(cma)
```

```
[[9.440e+02 0.000e+00 1.100e+01 0.000e+00 0.000e+00 2.000e+00 1.000e+01
  7.000e+00 5.000e+00 1.000e+00]
 [0.000e+00 1.112e+03 2.000e+00 3.000e+00 1.000e+00 4.000e+00 3.000e+00
  1.000e+00 9.000e+00 0.000e+00]
 [1.000e+01 6.000e+00 9.210e+02 1.200e+01 1.500e+01 3.000e+00 1.900e+01
  1.500e+01 2.600e+01 5.000e+00]
 [1.000e+00 1.000e+00 3.100e+01 8.620e+02 2.000e+00 7.200e+01 5.000e+00
  1.400e+01 1.200e+01 1.000e+01]
 [2.000e+00 3.000e+00 6.000e+00 2.000e+00 9.100e+02 1.000e+00 1.200e+01
  6.000e+00 4.000e+00 3.600e+01]
 [1.200e+01 3.000e+00 6.000e+00 2.900e+01 1.900e+01 7.680e+02 1.900e+01
  9.000e+00 2.100e+01 6.000e+00]
 [1.400e+01 3.000e+00 2.100e+01 2.000e+00 2.200e+01 2.800e+01 8.650e+02
  0.000e+00 3.000e+00 0.000e+00]
 [0.000e+00 1.400e+01 3.000e+01 9.000e+00 7.000e+00 2.000e+00 1.000e+00
  9.290e+02 3.000e+00 3.300e+01]
 [1.200e+01 1.600e+01 1.800e+01 2.600e+01 2.400e+01 4.600e+01 2.200e+01
  1.900e+01 7.720e+02 1.900e+01]
 [1.000e+01 4.000e+00 6.000e+00 2.200e+01 5.300e+01 1.800e+01 0.000e+00
  4.800e+01 4.000e+00 8.440e+02]]
```

calcul du recall et de la précision de chaque classe

```
def get_recall_and_precision(confusion_matrix, nb_class):
    tp_all = [confusion_matrix[i,i] for i in range(nb_class)]
    fn_all = [sum(confusion_matrix[i,:]) - tp_all[i] for i in range(nb_class)]
    fp_all = [sum(confusion_matrix[:,i]) - tp_all[i] for i in range(nb_class)]

    recall_all = [tp_all[i]/(tp_all[i] + fn_all[i]) for i in range(nb_class)]
    precision_all = [tp_all[i]/(tp_all[i] + fp_all[i]) for i in range(nb_class)]

    return recall_all,precision_all
```

sélection des meilleurs et pires classes en fonction du recall et de la précision

```
recall_all, precision_all = get_recall_and_precision(cma,nb_class_a)

print("Pire classes :")
print("Recall : {}".format(np.argsort(recall_all)[-1]))
print("Precision : {}".format(np.argsort(precision_all)[-1]))

print("Meilleure classes :")

print("Recall : {}".format(np.argsort(recall_all)[0]))
print("Precision : {}".format(np.argsort(precision_all)[0]))
```

```
Pire classes :
Recall : 1
Precision : 1
Meilleure classes :
Recall : 8
Precision : 5
```

calcul du F1 score

```
def get_f1_score(recall_all,precision_all,nb_class):
    f1_all = [2 * (precision_all[i]*recall_all[i])/
              (precision_all[i] + recall_all[i])
             for i in range(nb_class)]

    return np.mean(f1_all)

f1_a = get_f1_score(recall_all,precision_all,nb_class_a)
print("f1 score : {}".format(f1_a))
```

```
f1 score : 0.8907308492877297
```

```
data_b = np.loadtxt('ex2-system-b.csv', delimiter=';')

dtb_prior = data_b[:,:-1]
dtb_y = data_b[:,-1].astype(int)
nb_class_b = len(np.unique(dtb_y))

preds_b = get_preds(dtb_prior)

er_b = ovrl_error(dtb_y,preds_b)

cmb = get_confusion_matrix(dtb_y,preds_b,nb_class_b)

recall_all, precision_all = get_recall_and_precision(cmb, nb_class_b)
```

```
f1_b = get_f1_score(recall_all,precision_all,nb_class)
```

```
print("Comparaison des systemes :")
print("Systeme A :")
print("\toverall error rate : {}".format(er_a))
print("\tf1 score : {}".format(f1_a))
print("Systeme B :")
print("\toverall error rate : {}".format(er_b))
print("\tf1 score : {}".format(f1_b))
```

```
Comparaison des systemes :
Systeme A :
        overall error rate : 0.10729999999999995
        f1 score : 0.8907308492877297
Systeme B :
        overall error rate : 0.03869999999999996
        f1 score : 0.9608568150389065
```

Grâce à ces deux mesures on peut en conclure que le système B est un meilleur classifieur que le système A.