Rapport de projet de reconnaissance d'images

Lucas David & Théo Legars

1 Implémentation et utilisation du classificateur à distance minimum (DMIN)

On choisi d'implémenter ce classificateur sous forme d'une classe, ceci étant le plus courant et le plus pratique pour encapsuler les comportements et stocker les données nécessaires.

```
import numpy as np

class DMIN:
  def __init__(self):
    self.data = np.array([])
    self.label = np.array([])
    self.n_label = 0

  def fit(self, data, label):
    self.data = data
    self.label = label
    self.n_label = len(set(label))

  def predict(self, data):
    return [self.label[np.argmin(np.sum(np.subtract(self.data, data[iterator])
   ** 2, axis=1))] for iterator in range(0, len(data))]

  def score(self, data, label):
    return np.count_nonzero(self.predict(data) == label) / len(data)
```

L'utilisation se résumera aux lignes de code suivante :

```
from dmin import DMIN

dmin = DMIN()
dmin.fit(<data>, <label>)

# Prediction
dmin.score(<data>)

# Score
dmin.score(<data>, <label>)
```

On peut donc déterminer le taux de réussite via la fonction membre `score(<données à tester>, <labels correspondants>)`.

Dans le cas de nos données de développement, on obtient un score de 68,80% pour une exécution de 96,45 secondes. Sur l'ensemble d'entraînement il est intéressant de voir que le score n'est pas 100% mais moins,