**UPM – FCI – Ciência da Computação**

## Inteligência Artificial – Exercício de Programação no. 1

## 1o. semestre de 2018

**Em grupos de entre 4 (quatro) a 6 (seis) alunos, usando a biblioteca Scikit-Learn, realize as tarefas descritas abaixo. Submeta, via Moodle (uma única submissão por grupo!):**
**i) o nome e o TIA de todos os integrantes do grupo;**
**ii) o código em Python utilizado no desenvolvimento do exercício;**
**iii) print screen ou foto do código rodando no Anaconda-Navigator ou equivalente, incluindo código, figuras e demais saídas produzidas e**
**iv) as respostas às questões formuladas abaixo.**

**Data de entrega: 1º. de abril.**
**Grupos com número de integrantes diferentes dos descritos acima não terão suas submissões corrigidas e terão nota zero atribuídas às mesmas. A não ser que seu número de integrantes tenha sido previamente acertado com o professor.**
**Não será aceita a inclusão de nenhum membro ao grupo após a entrega.**

Scikit-Learn possui um dataset de dígitos cursivos. Cada dígito no dataset é representado por uma matriz de pixels binária, de dimensão 8x8.

Vamos importar esse dataset:

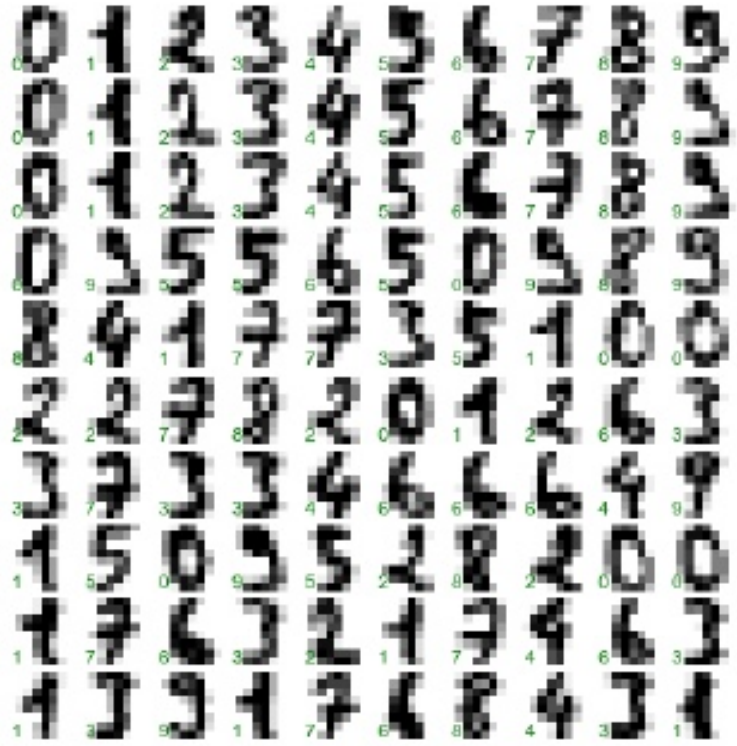
```
In:   from sklearn.datasets import load_digits
      digits = load_digits()
      digits.images.shape

Out:  (1797, 8, 8)
```

As imagens correspondentes aos dados correspondem a um array 3-D: 1,797 amostras, cada uma consistindo de uma matrix 8×8 de pixels. Vamos visualizar as primeiras 100 dessas imagens:

```
In:   import matplotlib.pyplot as plt

      fig, axes = plt.subplots(10, 10, figsize=(8, 8),
            subplot_kw={'xticks':[], 'yticks':[]},
            gridspec_kw=dict(hspace=0.1, wspace=0.1))
      for i, ax in enumerate(axes.flat):
            ax.imshow(digits.images[i], cmap='binary', interpolation='nearest')
            ax.text(0.05, 0.05, str(digits.target[i]),
                  transform=ax.transAxes, color='green')
```



O objetivo deste projeto é agrupar os exemplos no dataset. Agrupar, não classificar, que seria a tarefa típica sobre tal dataset. O que queremos investigar é até que ponto é possível – usando aprendizagem não supervisionada – identificar estrutura no dataset de modo a separa-lo em grupos correspondentes aos dígitos corretos. (Talvez não vamos conseguir sucesso absoluto; talvez sim, talvez não -- o experimento dirá.)

Sua **primeira tarefa** é agrupar os dígitos. Para isso você usará o algoritmo **k-means** da Scikit-Learn. **Preencha os trechos de código faltantes e rode o k-means**.

*In:* *from sklearn.cluster import KMeans*
*kmeans = **XXX**(n_clusters=**YYY**, random_state=0)*
*clusters = kmeans.fit_predict(digits.data)*
*kmeans.cluster_centers_.shape*

Para ver o que o agrupamento criou vamos olhar o centro de cada grupo (cluster) – o protótipo para cada grupo –, usando o código a seguir.

*In:* *fig, ax = plt.subplots(2, 5, figsize=(8, 3))*
*centers = kmeans.cluster_centers_.reshape(10, 8, 8)*
*for axi, center in zip(ax.flat, centers):*
*axi.set(xticks=[], yticks=[])*
*axi.imshow(center, interpolation='nearest', cmap=plt.cm.binary)*

Você também pode medir numericamente a correção do agrupamento, a partir do seguinte código:

```
In:     import numpy as np
        from scipy.stats import mode
        from sklearn.metrics import accuracy_score

        labels = np.zeros_like(clusters)
        for i in range(10):
                mask = (clusters == i)
                labels[mask] = mode(digits.target[mask])[0]
        accuracy_score(digits.target, labels)
```

Sua **segunda tarefa** é dizer, a partir da figura e do score obtidos acima:
a) **quais dígitos foram corretamente agrupados** (se algum) e
b) **quais dígitos foram erroneamente agrupados** (se algum).