TC1002S Herramientas computacionales: el arte de la analítica

This is a notebook with all your work for the final evidence of this course

Niveles de dominio a demostrar con la evidencia

SING0202A

Interpreta interacciones entre variables relevantes en un problema, como base para la construcción de modelos bivariados basados en datos de un fenómeno investigado que le permita reproducir la respuesta del mismo. Es capaz de construir modelos bivariados que expliquen el comportamiento de un fenómeno.

Student information

- Name: Manuel Alejandro Preciado Morán
- ID: A01639643
- My carreer: IRS

Importing libraries

In [ ]:

```
import numpy as np                        # For array
import pandas as pd                       # For data handling
import seaborn as sns                     # For advanced plotting
import matplotlib.pyplot as plt           # For showing plots
from sklearn.cluster import KMeans        # Import sklearn KMeans
from sklearn.datasets import load_digits
```

PART 1

Use your assigned dataset

A1 Load data

In [ ]:
```
# Dataset url
path = "/home/alex/TC1002S/NotebooksStudents/A01639643/Evidencia/A01639643.cs

# Load the dataset
df  = pd.read_csv(path)
df
```

Out[ ]:

	Unnamed: 0	x1	x2
0	0	-2.073678	0.085021
1	1	0.021777	0.353588
2	2	0.062024	0.361686
3	3	0.623642	0.939028
4	4	-1.922845	-0.140768
...	...	...	...
1995	1995	0.781006	0.407278
1996	1996	0.023259	1.090469
1997	1997	-0.873758	-0.536620
1998	1998	0.422635	0.707325
1999	1999	-0.642749	0.838029

2000 rows × 3 columns

A2 Data managment

Print the first 7 rows

In [ ]:
```
df.head(7)
```

Out[ ]:

	Unnamed: 0	x1	x2
0	0	-2.073678	0.085021
1	1	0.021777	0.353588
2	2	0.062024	0.361686
3	3	0.623642	0.939028
4	4	-1.922845	-0.140768
5	5	0.973537	-0.041355
6	6	-0.331686	-0.393125

Print the first 4 last rows

In [ ]:
```
df.tail(4)
```

Out[ ]:

	Unnamed: 0	x1	x2
1996	1996	0.023259	1.090469
1997	1997	-0.873758	-0.536620
1998	1998	0.422635	0.707325
1999	1999	-0.642749	0.838029

How many rows and columns are in your data?

Use the `shape` method

In [ ]:
```
print("The number of observations are: ", df.shape[0])
print("The number of variables are: ", df.shape[1])
```

```
The number of observations are:  2000
The number of variables are:  3
```

Print the name of all columns

Use the `columns` method

In [ ]:
```
df.columns
```

Out[ ]:
```
Index(['Unnamed: 0', 'x1', 'x2'], dtype='object')
```

What is the data type in each column

Use the `dtypes` method

In [ ]:
```
df.dtypes
```

Out[ ]:
```
Unnamed: 0      int64
x1            float64
x2            float64
dtype: object
```

What is the meaning of rows and columns?

In [ ]:
```
# Your responses here

# 1) The number of rows means the number of observations for each variable

# 2) The number of columns means the number of variables for the data set

# 3)

#...
```

Print a statistical summary of your columns

In [ ]:
```
df.describe()
```

Out[ ]:

	Unnamed: 0	x1	x2
count	2000.000000	2000.000000	2000.000000
mean	999.500000	-0.499319	0.251566
std	577.494589	0.869908	0.507518
min	0.000000	-2.209887	-0.756123
25%	499.750000	-1.087289	-0.199894
50%	999.500000	-0.498569	0.253424
75%	1499.250000	0.067590	0.691262
max	1999.000000	1.264445	1.275328

In [ ]:

```
# 1) What is the minumum and maximum values of each variable
print("The minimal value of each variable are: \n",df.min())
print("The maximum value of each variable are: \n",df.max())
# 2) What is the mean and standar deviation of each variable
print("The mean value of each variable are: \n",df.mean())
print("The standar deviation value of each variable are: \n",df.std())
# 3) What the 25%, 50% and 75% represent?
# They represent what value (between the min and the max) corresponds to the
# 50%, or 75% of the dataset
```

```
The minimal value of each variable are:
 Unnamed: 0    0.000000
x1           -2.209887
x2           -0.756123
dtype: float64
The maximum value of each variable are:
 Unnamed: 0    1999.000000
x1              1.264445
x2              1.275328
dtype: float64
The mean value of each variable are:
 Unnamed: 0    999.500000
x1             -0.499319
x2              0.251566
dtype: float64
The standar deviation value of each variable are:
 Unnamed: 0    577.494589
x1              0.869908
x2              0.507518
dtype: float64
```

Rename the columns using the same name with capital letters

In [ ]:

```
df = df.rename(columns={"Unnamed: 0":"0","x1":"X1", "x2":"X2"})
df
```

Out[ ]:

	0	X1	X2
0	0	-2.073678	0.085021
1	1	0.021777	0.353588

	0	X1	X2
2	2	0.062024	0.361686
3	3	0.623642	0.939028
4	4	-1.922845	-0.140768
...	...	...	...
1995	1995	0.781006	0.407278
1996	1996	0.023259	1.090469
1997	1997	-0.873758	-0.536620
1998	1998	0.422635	0.707325
1999	1999	-0.642749	0.838029

Rename the columns to their original names

In [ ]:

```
df = df.rename(columns={"X1":"x1", "X2":"x2"})
df
```

Out[ ]:

	0	x1	x2
0	0	-2.073678	0.085021
1	1	0.021777	0.353588
2	2	0.062024	0.361686
3	3	0.623642	0.939028
4	4	-1.922845	-0.140768
...	...	...	...
1995	1995	0.781006	0.407278
1996	1996	0.023259	1.090469
1997	1997	-0.873758	-0.536620
1998	1998	0.422635	0.707325
1999	1999	-0.642749	0.838029

2000 rows × 3 columns

Use two different alternatives to get one of the columns

In [ ]:

```
a = df.x1
b = df["x1"]

print(a)
print(b)
```

```
0      -2.073678
1       0.021777
2       0.062024
3       0.623642
4      -1.922845
```

```
          ...
1995    0.781006
1996    0.023259
1997   -0.873758
1998    0.422635
1999   -0.642749
Name: x1, Length: 2000, dtype: float64
0      -2.073678
1       0.021777
2       0.062024
3       0.623642
4      -1.922845
          ...
1995    0.781006
1996    0.023259
1997   -0.873758
1998    0.422635
1999   -0.642749
Name: x1, Length: 2000, dtype: float64
```

Get a slice of your data set: second and thrid columns and rows from 62 to 72

In [ ]:

```
df.iloc[62:73,1:3]
```

Out[ ]:

	x1	x2
62	-0.535739	0.724475
63	-0.197894	-0.195521
64	0.876373	0.661298
65	-1.860521	0.190419
66	-0.273754	0.985876
67	-0.185618	0.209379
68	-0.964107	0.437611
69	-0.027623	0.438405
70	-0.735647	0.542897
71	-0.476929	0.949102
72	-1.889235	0.504038

For the second and thrid columns, calculate the number of null and not null values and verify that their sum equals the total number of rows

In [ ]:

```
print(df.isnull().sum())
print(df.notnull().sum())
print(df.count())
```

```
0     0
x1    0
x2    0
dtype: int64
0     2000
x1    2000
```

```
x2    2000
dtype: int64
0     2000
x1    2000
x2    2000
dtype: int64
```

Discard the first/last column

In [ ]:

```
df.drop(columns=["0"], inplace=True)
df
```

Out[ ]:

	x1	x2
0	-2.073678	0.085021
1	0.021777	0.353588
2	0.062024	0.361686
3	0.623642	0.939028
4	-1.922845	-0.140768
...	...	...
1995	0.781006	0.407278
1996	0.023259	1.090469
1997	-0.873758	-0.536620
1998	0.422635	0.707325
1999	-0.642749	0.838029

2000 rows × 2 columns

Questions

Based on the previos results, provide a description of yout dataset

Your response:

Now that we have apply some changes to the dataset, we can read it more easily, due to the remove of redundant information(the index). Also, we can see that we have 2000 observations (rows) for each variable, and 2 variables (columns)

A3 Data visualization

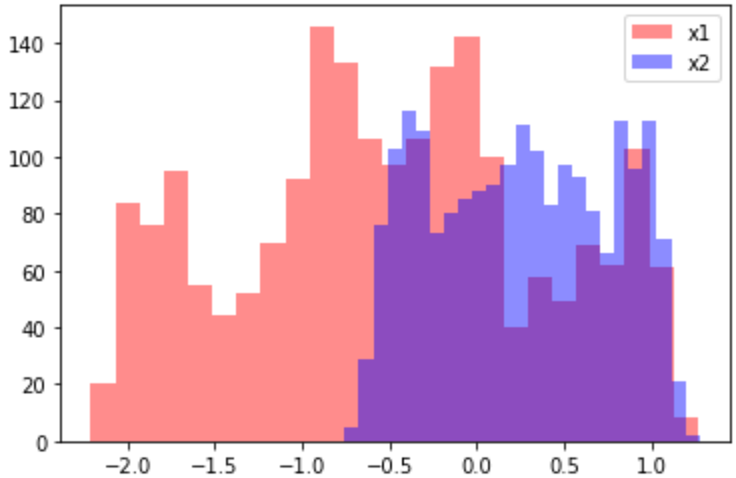
Plot in the same figure the histogram of the two variables

In [ ]:

```
plt.hist(df['x1'], bins=25, alpha=0.45, color='red')
plt.hist(df['x2'], bins=25, alpha=0.45, color='blue')
plt.legend(['x1', 'x2'])
plt.show
```

Out[ ]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Based on this plots, provide a description of your data:

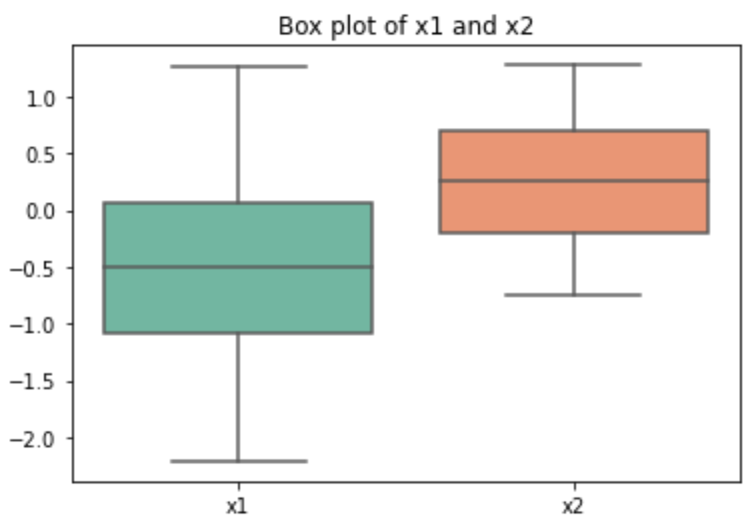
Your response here: We can see that the values of x1 goes all the way from -2.1 to 1.2. And that the highlights of this variables are in -1.0 and -0.1, those are the values that appear more often.

Now, for the x2 variable, we can see that the values of this variable only comes frome -0.8 to 1.2

Plot in the same figure the boxplot of the two variables

In [ ]:

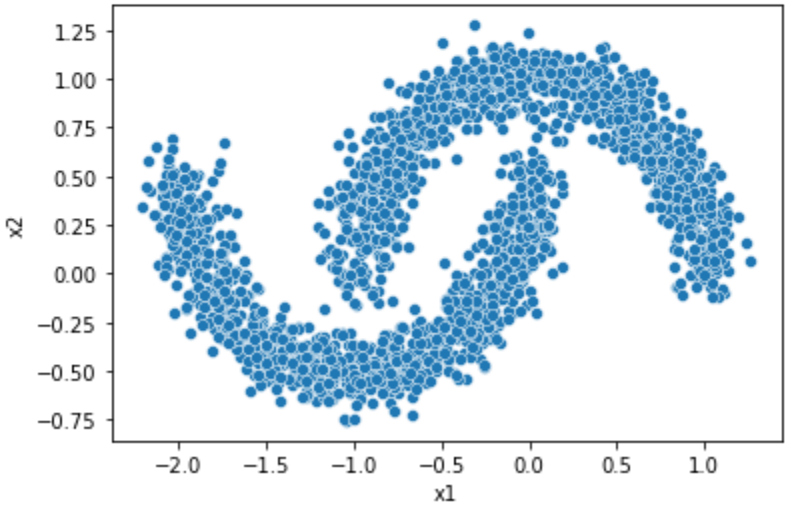
```
x = df.loc[:,["x1", "x2"]]
x2bp = sns.boxplot(data = x, orient = "v", palette = "Set2")
x2bp.set_title("Box plot of x1 and x2")
plt.show()
```



Scatter plot of the two variables

In [ ]:

```
sns.scatterplot(data = df, x = "x1", y = "x2")
plt.show()
```



Questions

Based on the previos plots, provide a description of yout dataset

Your response: The boxplot gives us an idea of which values repeat the most. Meanwhile, in the scatter plot we can watch the relation between both of the variables. Also, at first I was thinking we could watch 2 clusters, due to we have 2 half moons. But afterwards, it came to mind that 4 clusters could be better, so we could slice each half moon into 2.

A4 Kmeans

Do Kmeans clustering assuming a number of clusters accorging to your scatter plot

In [ ]:

```
# Define number of clusters
K  =  6# Let's assume there are 2,3,4,5...? clusters/groups

# Create the Kmeans box/object
km = KMeans(n_clusters = K)

# Do K-means clustering (assing each point in the dataset to a cluster)
yestimated = km.fit_predict(df)


# Print estimated cluster of each point in the dataset
yestimated
```

Out[ ]:

```
array([4, 3, 3, ..., 1, 5, 0], dtype=int32)
```

Add to your dataset a column with the assihned cluster to each data point

In [ ]:

```
df['yestimated'] = yestimated
```

Print the number associated to each cluster

In [ ]:

```
df
```

Out[ ]:

	x1	x2	yestimated
0	-2.073678	0.085021	4
1	0.021777	0.353588	3
2	0.062024	0.361686	3
3	0.623642	0.939028	5
4	-1.922845	-0.140768	4
...	...	...	...
1995	0.781006	0.407278	2
1996	0.023259	1.090469	5
1997	-0.873758	-0.536620	1
1998	0.422635	0.707325	5
1999	-0.642749	0.838029	0

2000 rows × 3 columns

In [ ]:

```
df.drop(columns=["yestimated"], inplace=True)
df
```

Out[ ]:

	x1	x2
0	-2.073678	0.085021
1	0.021777	0.353588
2	0.062024	0.361686
3	0.623642	0.939028
4	-1.922845	-0.140768
...	...	...
1995	0.781006	0.407278
1996	0.023259	1.090469
1997	-0.873758	-0.536620
1998	0.422635	0.707325
1999	-0.642749	0.838029

2000 rows × 2 columns

Print the centroids

In [ ]:

```
# Cluster centroides
km.cluster_centers_
```

Out[ ]:

```
array([[-0.7674266 ,  0.57187726],
       [-1.07640045, -0.41353825],
       [ 0.8566732 ,  0.43120725],
```

```
       [-0.2169569 , -0.05806358],
       [-1.84577581,  0.06021696],
       [ 0.08727478,  0.91667382]])
```

Print the intertia metric

In [ ]:

```python
# Sum of squared error (sse) of the final model
km.inertia_
```

Out[ ]: 197.28858067486814

Plot a scatter plot of your data assigned to each cluster. Also plot the centroids

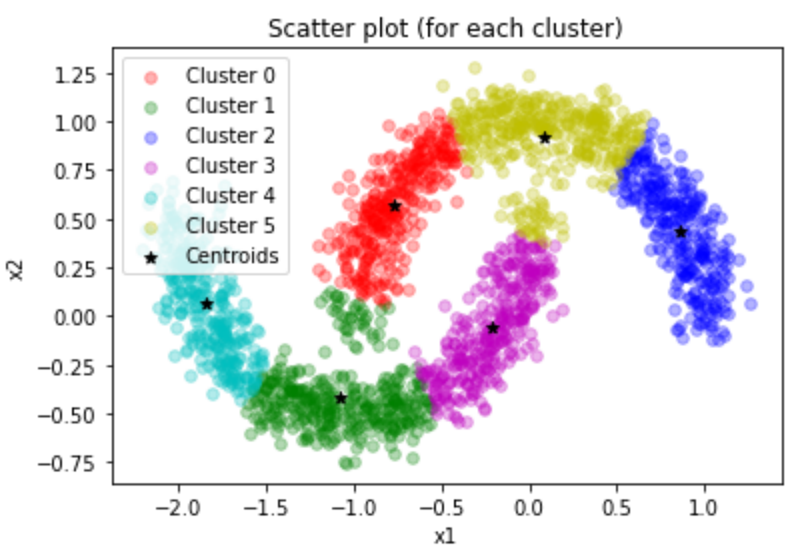
In [ ]:

```python
# Get a dataframe with the data of each cluster
df1 = df[yestimated==0]
df2 = df[yestimated==1]
df3 = df[yestimated==2]
df4 = df[yestimated==3]
df5 = df[yestimated==4]
df6 = df[yestimated==5]

# Scatter plot of each cluster
plt.scatter(df1.x1, df1.x2, label="Cluster 0", c="r", marker = "o", s=32, alp
plt.scatter(df2.x1, df2.x2, label="Cluster 1", c="g", marker= "o", s=32, alph
plt.scatter(df3.x1, df3.x2, label="Cluster 2", c="b", marker= "o", s=32, alph
plt.scatter(df4.x1, df4.x2, label="Cluster 3", c="m", marker = "o", s=32, alp
plt.scatter(df5.x1, df5.x2, label="Cluster 4", c="c", marker= "o", s=32, alph
plt.scatter(df6.x1, df6.x2, label="Cluster 5", c="y", marker= "o", s=32, alph

#Plot centroids
plt.scatter(km.cluster_centers_[:,0], km.cluster_centers_[:,1], color='black'

plt.title("Scatter plot (for each cluster)")
plt.xlabel("x1")
plt.ylabel("x2")
plt.legend()
plt.show()
```



Questions

Provides a detailed description of your results

Your response: In the previous graph we can see that indeed the data can be devided into 6 clusters with each of its centroids. Maybe if we added more clusters into the scatter plot the analisis could respect the data from the upper or lower half moon.

A5 Elbow plot

Compute the Elbow plot

In [ ]:

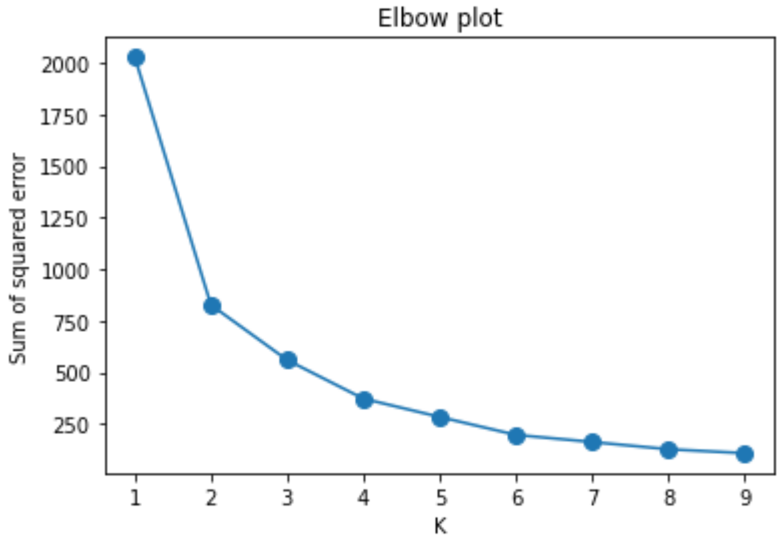
```
# Intialize a list to hold sum of squared error (sse)
sse = []

# Define values of k
k_rng = range(1,10)

# For each k
for k in k_rng:
    #Create model
    km = KMeans(n_clusters=k)
    #Do K-means clustering
    km.fit_predict(df[["x1", "x2"]])
    #Save sse for each k
    sse.append(km.inertia_)
```

In [ ]:

```
# Plot sse versus k
plt.plot(k_rng,sse, "o-", markersize=8)
plt.title("Elbow plot")
plt.xlabel("K")
plt.ylabel("Sum of squared error")
plt.show()
```



Questions

What is the best number of clusters K? (argue your response)

Your response: I think that the best number of clusters K is 6, due to prior to that numer we can see a significant change between each number. But after that we can see that the change is minor.

Does this number of clusters agree with your inital guess? (argue your response)

Your response: No, my first guess was 2 or 4 clusters, due to the half moons we got in the scatter plot

PART 2

Load and do clustering using the "digits" dataset

1) Load the dataset using the "load_digits()" function from "sklearn.datasets"

In [ ]:

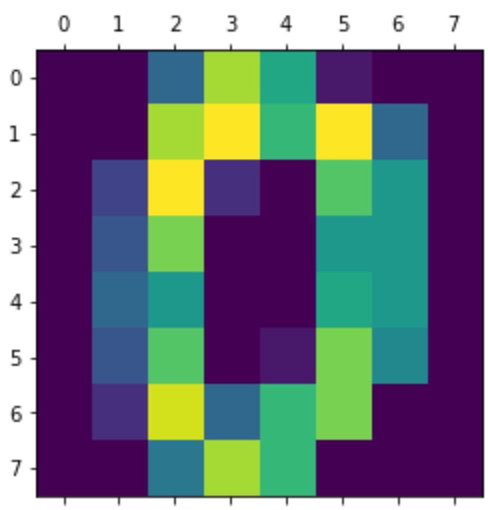
```
digits = load_digits()
digits.data.shape
```

Out[ ]: (1797, 64)

2) Plot some of the observations

In [ ]:

```
plt.matshow(digits.images[0])
plt.show()
```



In [ ]:

```
d = pd.DataFrame(digits["data"])
d
```

Out[ ]:

0	1	2	3	4	5	6	7	8	9	...	54	55	56	57	58	59	60	61

	0	1	2	3	4	5	6	7	8	9	...	54	55	56	57	58	59	60	61
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	6.0	13.0	10.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	11.0	16.0	10.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	5.0	0.0	0.0	0.0	0.0	3.0	11.0	16.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	9.0	0.0	0.0	0.0	7.0	13.0	13.0	9.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	2.0	16.0	4.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1792	0.0	0.0	4.0	10.0	13.0	6.0	0.0	0.0	0.0	1.0	...	4.0	0.0	0.0	0.0	2.0	14.0	15.0	9.0
1793	0.0	0.0	6.0	16.0	13.0	11.0	1.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	6.0	16.0	14.0	6.0
1794	0.0	0.0	1.0	11.0	15.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	2.0	9.0	13.0	6.0
1795	0.0	0.0	2.0	10.0	7.0	0.0	0.0	0.0	0.0	0.0	...	2.0	0.0	0.0	0.0	5.0	12.0	16.0	12.0
1796	0.0	0.0	10.0	14.0	8.0	1.0	0.0	0.0	0.0	2.0	...	8.0	0.0	0.0	1.0	8.0	12.0	14.0	12.0

3) Do K means clustering

In [ ]:

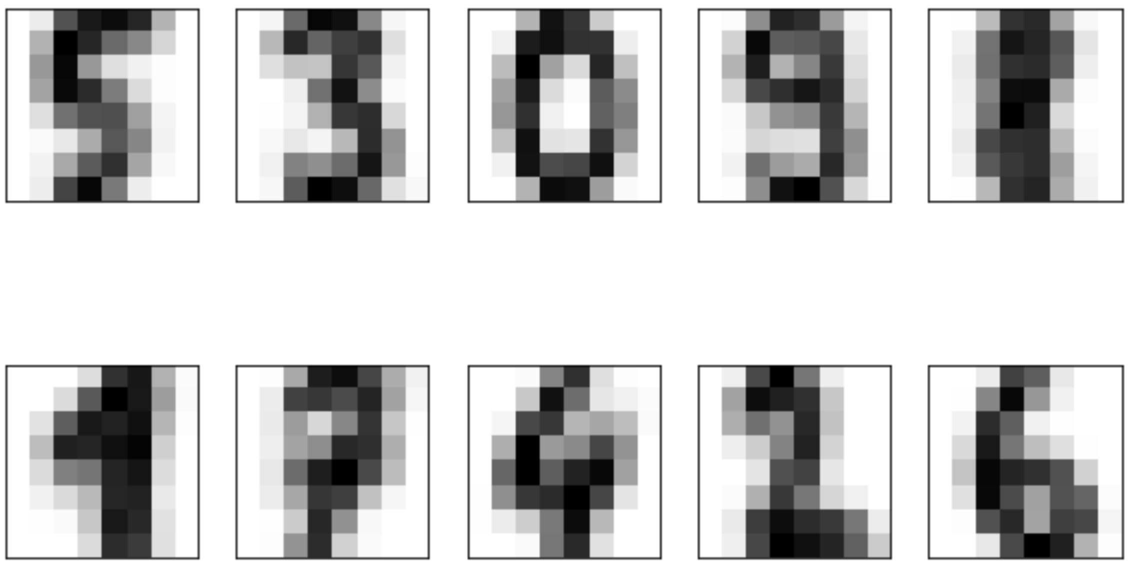
```
# Define number of clusters
K  =  10# Let's assume there are 2,3,4,5...? clusters/groups

# Create the Kmeans box/object
km = KMeans(n_clusters = K)

# Do K-means clustering (assing each point in the dataset to a cluster)
yestimated = km.fit_predict(d)

# Print estimated cluster of each point in the dataset
km.cluster_centers_.shape

fig, ax = plt.subplots(2, 5, figsize=(10, 6))
centers = km.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)
```

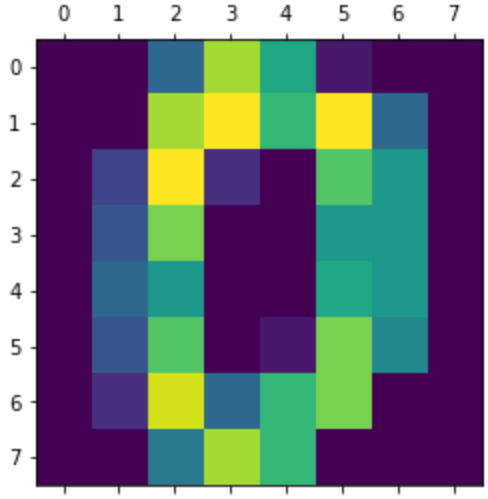


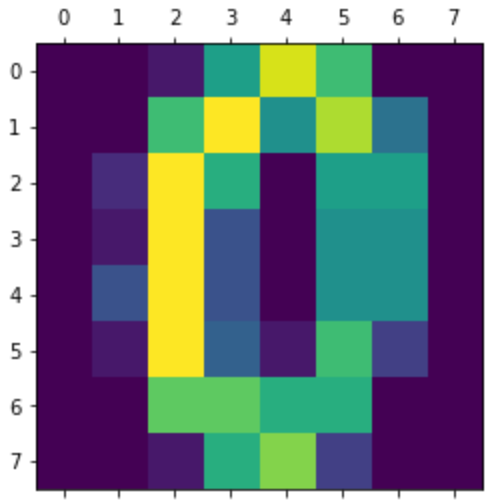
4) Verify your results in any of the observations

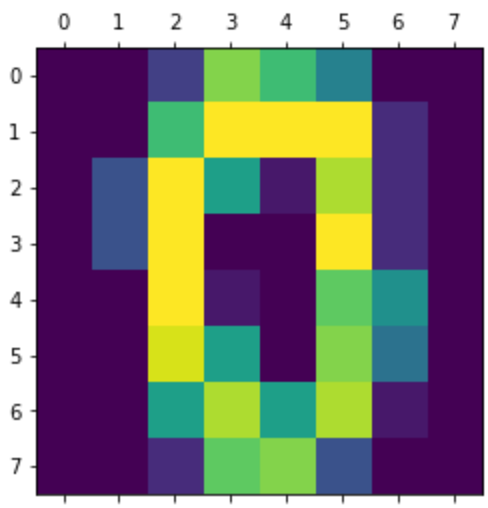
In [ ]:

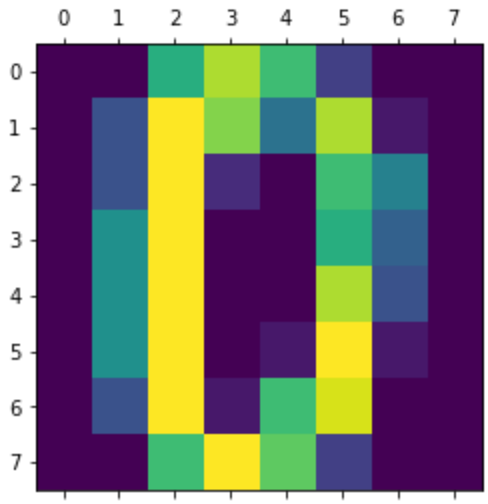
```
plt.matshow(digits.images[0])
print(yestimated[0])
plt.matshow(digits.images[10])
print(yestimated[10])
plt.matshow(digits.images[20])
print(yestimated[20])
plt.matshow(digits.images[30])
print(yestimated[30])
plt.matshow(digits.images[48])
print(yestimated[48])
```

```
2
2
2
2
2
```


0 1 2 3 4 5 6 7
0 1 2 3 4 5 6 7


0 1 2 3 4 5 6 7
0 1 2 3 4 5 6 7


0 1 2 3 4 5 6 7
0 1 2 3 4 5 6 7



Questions

Provides a detailed description of your results.

Your response: This part catalogs the images according to the centers created to each one of them. To check if it is being catloged the correct way, we used the yestimated of different zeros located in the dataset. Once you've done that you can see if the algorithm is working correctly.

PART 3

Descipcion de tu percepcion del nivel de desarrollo de la subcompetencia

SING0202A Interpretación de variables

Escribe tu description del nivel de logro del siguiente criterio de la subcompetencia

Interpreta interacciones. Interpreta interacciones entre variables relevantes en un problema, como base para la construcción de modelos bivariados basados en datos de un fenómeno investigado que le permita reproducir la respuesta del mismo.

Tu respuesta: A lo largo de esta semana hemos analizado e interpretado datos de diferentes fenómenos investigados para el desarrollo de nuevas habilidades. Dichos datos constan de variables interconectadas las cuales inicialimos como un dataset para después realizar los gráficos pertinentes. A través de las diferentes actividades realizadas considero que el nivel de comprensión para el análisis de datos es alto y puede ser aplicado en futuros proyectos.

Escribe tu description del nivel de logro del siguiente criterio de la subcompetencia

Construcción de modelos. Es capaz de construir modelos bivariados que expliquen el comportamiento de un fenómeno.

Tu respuesta:

Considero que con las diferentes prácticas se han adquirido una serie de habilidades las cuales nos permitirán construir modelos futuros a partir de datos de diferentes proyectos.