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: Andres Benjamin Antelis Moreno
- ID: A01637683
- My carreer: ITC

Importing libraries

```
# Import the packages that we will be using
import pandas as pd                 # For data handling
import seaborn as sns               # For advanced plotting
import matplotlib.pyplot as plt     # For showing plots

# Note: specific functions of the "sklearn" package will be imported when needed to show concepts easily
```

PART 1

Use your assigned dataset

A1 Load data

```
from google.colab import drive
drive.mount('/content/drive')
```

```
     Mounted at /content/drive
```

```
df  = pd.read_csv("/content/drive/MyDrive/!Tec stuff/!Uni/Semestre 2/Semana Tec1/A01637683.csv")

df
```

	Unnamed: 0	x1	x2
0	0	0.469817	0.061671
1	1	-0.440380	0.368456

A2 Data managment

Print the first 7 rows

...	...	...	...

```
df.head(7)
```

	Unnamed: 0	x1	x2
0	0	0.469817	0.061671
1	1	-0.440380	0.368456
2	2	-0.160479	1.010510
3	3	-0.671303	-0.761080
4	4	-0.416888	0.921896
5	5	-0.338697	-0.508942
6	6	-0.316442	-0.424257

Print the first 4 last rows

```
df.tail(4)
```

	Unnamed: 0	x1	x2
1020	1020	0.536685	-0.048738
1021	1021	-0.250926	-1.066358
1022	1022	-0.405477	1.004255
1023	1023	0.638617	-0.825212

How many rows and columns are in your data?

Use the `shape` method

```
df.shape
```

```
(1024, 3)
```

```
print("It has 1024 rows and 3 columns")
```

```
It has 1024 rows and 3 columns
```

Print the name of all columns

Use the `columns` method

```
df.columns
```

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

What is the data type in each column

Use the `dtypes` method

```
df.dtypes
```

```
Unnamed: 0      int64
x1            float64
x2            float64
dtype: object
```

What is the meaning of rows and columns?

```
# Your responses here

#The number of columns represent the amount of variables that we are analyzing in our analysis, and the number of rows represent the amount o
#variable
```

Print a statistical summary of your columns

```
# Summary statistics for the quantitative variables
df.describe()
```

	Unnamed: 0	x1	x2
count	1024.00000	1024.000000	1024.000000
mean	511.50000	-0.001341	-0.000723
std	295.74764	0.567212	0.564165
min	0.00000	-1.158472	-1.124957
25%	255.75000	-0.444466	-0.426706
50%	511.50000	0.000924	-0.001904
75%	767.25000	0.430471	0.437799
max	1023.00000	1.192867	1.173127

```
# 1) What is the minumum and maximum values of each variable
#First variable, Minimum:0, Maximum:1023
#Second variable, Minimum: -1.158472, Maximum: 1.19867
#Third variable, Minimum: -1.124957, Maximum: 1.173127

# 2) What is the mean and standar deviation of each variable
#First variable, Mean: 511.5 , Std: 295.74764
#Second variable, Mean: -0.001341, Std: 0.567212
#Third variable, Mean: -0.000723 , Std: 0.564165


# 3) What the 25%, 50% and 75% represent?
  #The 25%, 50% and 75% represent the percentiles, this means that the amount of data that is located with respect of each other, for example
  #higher than 80% of the number of data
```

Rename the columns using the same name with capital letters

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

	Index	X1	X2
0	0	0.469817	0.061671
1	1	-0.440380	0.368456
2	2	-0.160479	1.010510
3	3	-0.671303	-0.761080
4	4	-0.416888	0.921896

Rename the columns to their original names

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

df.head()
```

	Unnamed: 0	x1	x2
0	0	0.469817	0.061671
1	1	-0.440380	0.368456
2	2	-0.160479	1.010510
3	3	-0.671303	-0.761080
4	4	-0.416888	0.921896

Use two different alternatives to get one of the columns

```
a = df.x1
b = df["x2"]
```

```
a
```

```
0       0.469817
1      -0.440380
2      -0.160479
3      -0.671303
4      -0.416888
          ...
1019    0.526813
1020    0.536685
1021   -0.250926
1022   -0.405477
1023    0.638617
Name: x1, Length: 1024, dtype: float64
```

```
b
```

```
0       0.061671
1       0.368456
2       1.010510
3      -0.761080
4       0.921896
          ...
1019    0.829925
1020   -0.048738
1021   -1.066358
1022    1.004255
1023   -0.825212
Name: x2, Length: 1024, dtype: float64
```

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

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

	x1	x2
62	0.341460	-1.098711

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

```
print(df.x1.notnull().sum())
print(pd.isnull(df.x1).sum())

print(df.x2.notnull().sum())
print(pd.isnull(df.x2).sum())
```

```
1024
0
1024
0
```

The sum of the not null items is 1024, which means there are no null values

Discard the Index

```
df = df.rename(columns={"Unnamed: 0": "Index"})

df.drop("Index", axis=1, inplace = True)
```

```
df.head()
```

	x1	x2
0	0.469817	0.061671
1	-0.440380	0.368456
2	-0.160479	1.010510
3	-0.671303	-0.761080
4	-0.416888	0.921896

Questions

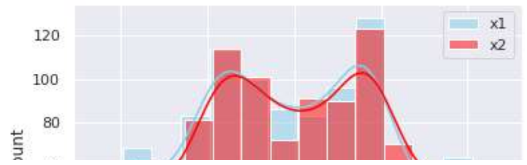
Based on the previos results, provide a description of yout dataset

Your response: In this dataset we have 2 quantitative variables, x1 & x2, that are receiving 1024 amount of data each, the 1024 refers to the number of observations that is measured for each variable, there are no null values for any variable, and the Index column which is the first one, just refers to the index number for each of the observations

A3 Data visualization

Plot in the same figure the histogram of the two variables

```
sns.set(style="darkgrid")

sns.histplot(data=df, x="x1", color="skyblue", label="x1", kde=True)
sns.histplot(data=df, x="x2", color="red", label="x2", kde=True)

plt.legend()
plt.show()
```

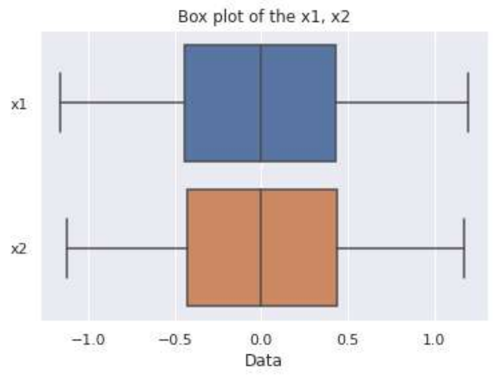


Based on this plots, provide a description of your data:

Your response here: As we can see from the plot, this tells us that both variables x1, and x2 have similar amount of data, this means that for each variable value, both the x1 and x2 the amount of data in the same interval will be similar in repetition

Plot in the same figure the boxplot of the two variables

```
BillAndTip = df.loc[:, ["x1", "x2"]]

x2bp = sns.boxplot(data=BillAndTip, orient="h")
x2bp.set_xlabel("Data")
x2bp.set_title("Box plot of the x1, x2")

plt.show()
```

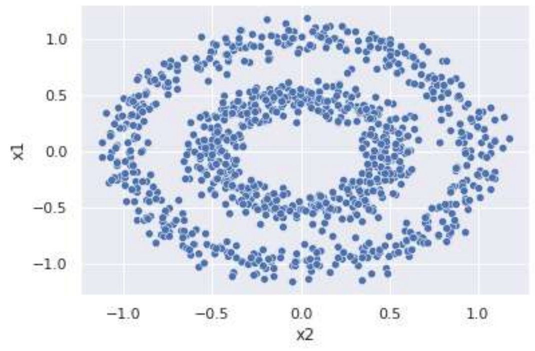


```
df.mean()
```

```
     x1   -0.001341
     x2   -0.000723
     dtype: float64
```

Scatter plot of the two variables

```
# scatter plot between two variables
sns.scatterplot(data=df, y="x1", x="x2")

plt.show()
```



Questions

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

Your response:Based on the boxplots, we can see that the range of data from both variables is extremely similar, we can see that the line of the mean is almost at the same place, however due to minimal differences the line is different from each, but both the range of data and the percentiles of both x1 and x2 is extremely similar, that is why the boxplot appears to be almost the same. And as for the scatter plot we can see how the variables relate to each other, and as we can see the data is formed in such a way that we can see 2 clusters of grouped data, one inside of another

A4 Kmeans

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

```
#Reset dataframe to experiment with results
df  = pd.read_csv("/content/drive/MyDrive/!Tec stuff/!Uni/Semestre 2/Semana Tec1/A01637683.csv")

df = df.rename(columns={"Unnamed: 0": "Index"})
df.drop("Index", axis=1, inplace = True)

df.head()
```

	x1	x2
0	0.469817	0.061671
1	-0.440380	0.368456
2	-0.160479	1.010510
3	-0.671303	-0.761080
4	-0.416888	0.921896

```
# Import sklearn KMeans
from sklearn.cluster import KMeans

# Define number of clusters
km = KMeans(n_clusters=3)

# Do K-means clustering (assing each point in the dataset to a cluster)
yp = km.fit_predict(df[['x1','x2']])

# Print estimated cluster of each point in the dataser
yp
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10
  warnings.warn(
array([2, 0, 0, ..., 1, 0, 2], dtype=int32)
```

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

```
df.insert(2, "yp", yp, True)

df.head()
```

	x1	x2	yp
0	0.469817	0.061671	0
1	-0.440380	0.368456	1
2	-0.160479	1.010510	1
3	-0.671303	-0.761080	2
4	-0.416888	0.921896	1

Print the number associated to each cluster

```
df.yp.unique()
```

```
array([0, 1, 2], dtype=int32)
```

Print the centroids

```
km.cluster_centers_
```

```
array([[ 0.6246084 ,  0.04448667],
       [-0.35971867,  0.51320057],
       [-0.2726145 , -0.5713359 ]])
```

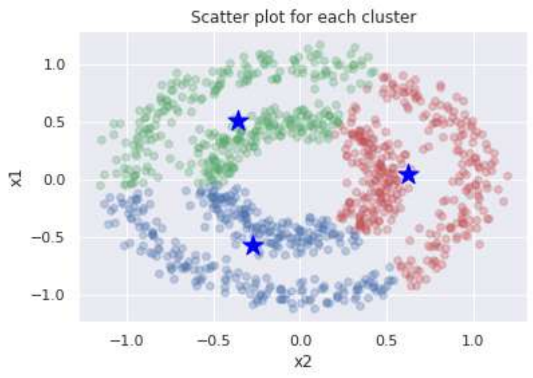
Print the intertia metric

```
km.inertia_
```

```
250.07078909960862
```

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

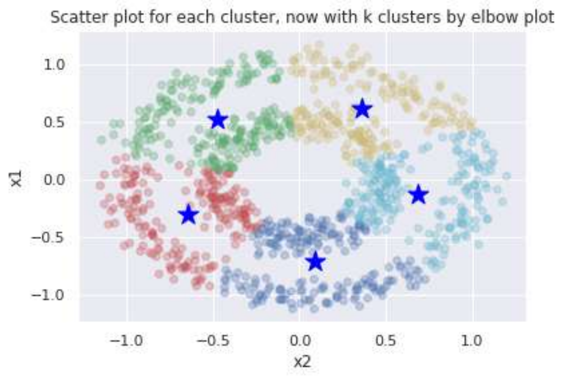
```
# Get a dataframe with the data of each clsuter
df1=df[df.yp==0]
df2=df[df.yp==1]
df3=df[df.yp==2]
#df4=df[df.yp==3]

# Scatter plot of each cluster
plt.scatter(df1.x1,df1.x2,label="cluster 0", c="r",s=32,alpha=0.3)
plt.scatter(df2.x1,df2.x2,label="cluster 0", c="g",s=32,alpha=0.3)
plt.scatter(df3.x1,df3.x2,label="cluster 0", c="b",s=32,alpha=0.3)
#plt.scatter(df4.x1,df4.x2,label="cluster 0", c="b",s=32,alpha=0.3)

plt.scatter(km.cluster_centers_[:,0], km.cluster_centers_[:,1], color='blue', marker='*', label='Centroides', s=256)

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



```
df  = pd.read_csv("/content/drive/MyDrive/!Tec stuff/!Uni/Semestre 2/Semana Tec1/A01637683.csv")

df = df.rename(columns={"Unnamed: 0": "Index"})
df.drop("Index", axis=1, inplace = True)

km = KMeans(n_clusters=5)

# Do K-means clustering (assing each point in the dataset to a cluster)
yp2 = km.fit_predict(df[['x1','x2']])

df.insert(2, "yp", yp2, True)

km.cluster_centers_

# Get a dataframe with the data of each clsuter
df1=df[df.yp==0]
df2=df[df.yp==1]
df3=df[df.yp==2]
df4=df[df.yp==3]
```

```
df5=df[df.yp==4]

# Scatter plot of each cluster
plt.scatter(df1.x1,df1.x2,label="cluster 0", c="r",s=32,alpha=0.3)
plt.scatter(df2.x1,df2.x2,label="cluster 0", c="g",s=32,alpha=0.3)
plt.scatter(df3.x1,df3.x2,label="cluster 0", c="b",s=32,alpha=0.3)
plt.scatter(df4.x1,df4.x2,label="cluster 0", c="c",s=32,alpha=0.3)
plt.scatter(df5.x1,df5.x2,label="cluster 0", c="y",s=32,alpha=0.3)

plt.scatter(km.cluster_centers_[:,0], km.cluster_centers_[:,1], color='blue', marker='*', label='Centroides', s=256)

plt.title("Scatter plot for each cluster, now with k clusters by elbow plot")
plt.xlabel("x2")
plt.ylabel("x1")
plt.show()
```

```
    /usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10
      warnings.warn(
```



Questions

Provides a detailed description of your results

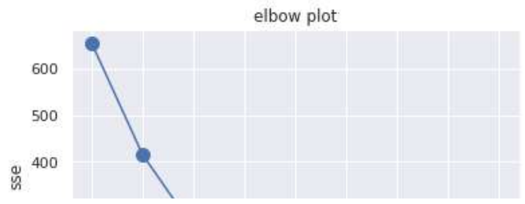
Your response: as we can see, the initial tought process that I had, which was 3 clusters wasnt efficient, we could sort the numbers better as we can see in the plot of 5 clusters, which was the most efficient way to sort them

A5 Elbow plot

Compute the Elbow plot

```
# 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:
  #creating the model
  km=KMeans(n_clusters=k,n_init="auto")
  #Do k means clustering
  km.fit_predict(df[["x1","x2"]])
  #saving for each k sse
  sse.append(km.inertia_)


# Plot sse versus k
plt.plot(k_rng,sse,"o-",markersize=10)
plt.title("elbow plot")
plt.xlabel("k")
plt.ylabel("sse")
```

```
Text(0, 0.5, 'sse')
```



Questions

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

Your response: According to the elbow plot, the best amount of clusters k will be 5, this is because the difference from 4 clusters to 5 clusters is low compared to the difference from 1 k to 4k, and we shouldnt choose more clusters like 9, even if the difference from 9 to 8 is lower, because now the amount of difference will be completely insignificant, and we could just choose 5 clusters

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

Your response: It does not, I choose 3 clusters, and we can do even more clusters if possible, we will be able to choose 5 clusters due to the elbow plot being able to tell us what is a good amount of clusters

PART 2

Load and do clustering using the "digits" dataset

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

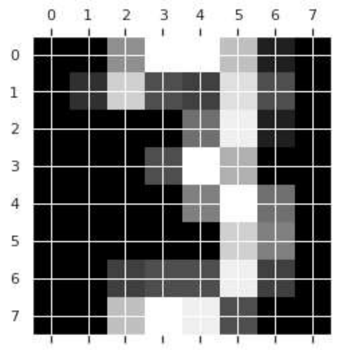
```
from sklearn.datasets import load_digits
digits = load_digits()

digits.keys()
print(digits.data.shape)
```

```
(1797, 64)
```

2) Plot some of the observations

```
plt.gray()
plt.matshow(digits.images[1300])
plt.show()
```

```
<Figure size 432x288 with 0 Axes>
```



3) Do K means clustering

```
digits.data
```

```
array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ..., 10.,  0.,  0.],
       [ 0.,  0.,  0., ..., 16.,  9.,  0.],
       ...,
       [ 0.,  0.,  1., ...,  6.,  0.,  0.],
           [ 0.,  0.,  2., ..., 12.,  0.,  0.],
           [ 0.,  0., 10., ..., 12.,  1.,  0.]])
```

```
df = pd.DataFrame(digits.data, columns = ['Column_A','Column_B','Column_C','Column_C','Column_C','Column_C','Column_C','Column_C','Column_C',
df.head()
```

	Column_A	Column_B	Column_C	Column_C	Column_C	Column_C	Column_C	Column_C
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0

5 rows × 64 columns

```
for i in range(0,64):
  print(i, end="")
```

```
0123456789101112131415161718192021222324252627282930313233343536373839404142434445464748495051525354555657585960616263
```

```
km = KMeans(n_clusters=10)

# Do K-means clustering (assing each point in the dataset to a cluster)
yp = km.fit_predict(df[['Column_A','Column_B','Column_C','Column_C','Column_C','Column_C','Column_C','Column_C','Column_C','Column_C','Column

# Print estimated cluster of each point in the dataser
yp
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10
  warnings.warn(
array([3, 4, 4, ..., 4, 8, 8], dtype=int32)
```

```
df.insert(64, "yp", yp, True)

df.head()
```

	Column_A	Column_B	Column_C	Column_C	Column_C	Column_C	Column_C	Column_C
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0

5 rows × 65 columns

```
df.yp.unique()
```

```
array([0, 2, 1], dtype=int32)
```

```
km.cluster_centers_
```

```
array([[0.00000000e+00, 8.01603206e-03, 2.03807615e+00, ...,
        7.43486974e+00, 1.79759519e+00, 8.61723447e-02],
       [0.00000000e+00, 5.25190840e-01, 8.13893130e+00, ...,
        1.01404580e+01, 3.92671756e+00, 9.23664122e-01],
       [0.00000000e+00, 3.07931571e-01, 4.67340591e+00, ...,
        2.80404355e+00, 3.84136858e-01, 1.08864697e-02]])
```

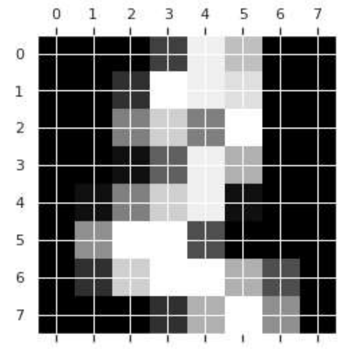
```
km.inertia_
```

```
107185757.88339198
```

4) Verify your results in any of the observations

```
i=2
plt.gray()
plt.matshow(digits.images[i])
plt.show()
print(df.iloc[i, 64])
```

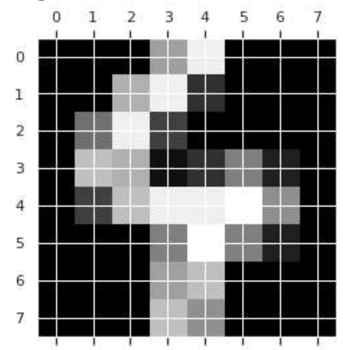
```
<Figure size 432x288 with 0 Axes>
```



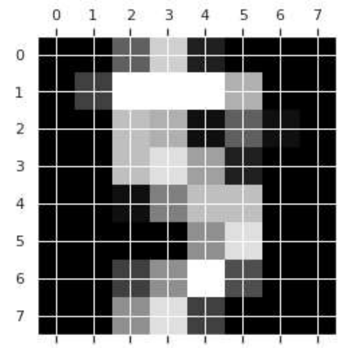
```
4
```

```
for i in range(110,210,10):
  plt.gray()
  plt.matshow(digits.images[i])
  plt.show()
  print("Print the cluster of image ",i ,"is ",df.iloc[i, 64], end="")
```

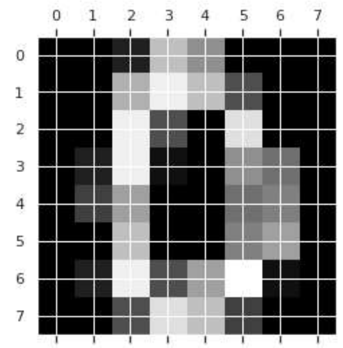
```
<Figure size 432x288 with 0 Axes>
```



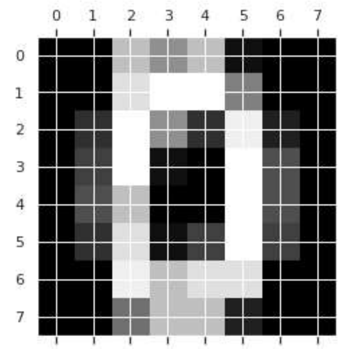
```
Print the cluster of image  110 is  6
<Figure size 432x288 with 0 Axes>
```



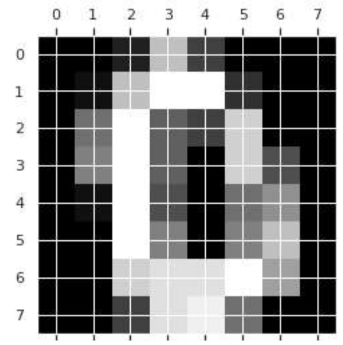
```
Print the cluster of image  120 is  0
<Figure size 432x288 with 0 Axes>
```



```
Print the cluster of image  130 is  3
<Figure size 432x288 with 0 Axes>
```



```
Print the cluster of image  140 is  3
<Figure size 432x288 with 0 Axes>
```



Questions

Provides a detailed description of your results.

Your response: As we can see here, the model assigns a cluster to each different number, and it analyzes the content of the array in the data frame and determines in which cluster it will be, in this case the cluster isnt correctely assigned to each number, however as we can see it sometimes gets the cluster equal in similar numbers, but sometimes it gets them wrong

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

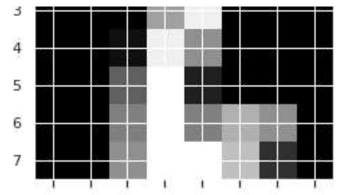
Interpreta interacciones. Interpreta interacciones entre variables relevantes en un problema, como base para la construcción de modelos bivariados basados en datos de un fenómeno investigado que le permita reproducir la respuesta del mismo.

Tu respuesta: Reconocer cuales son las variables importantes en un dataframe, y conocer como afectan estas en relacion a otras para de esta forma saber como crear plots para de esta misma manera poder crear graficas que nos ayuden a analyzar los datos y sacar conclusiones

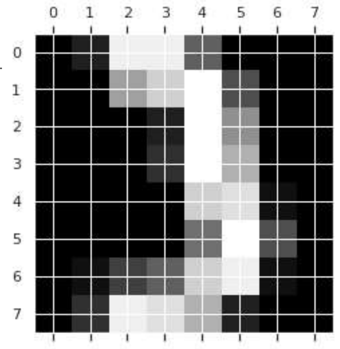
Escribe tu description del nivel de logro del siguiente criterio de la subcompetencia

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

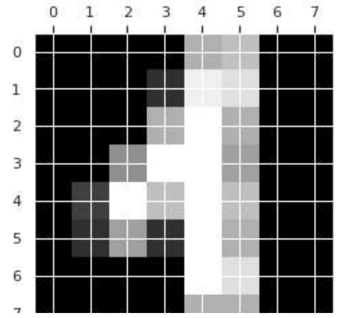
Tu respuesta: Saber como analizar y concluir un dataframe para de esta forma poder construir y calcular un modelo de k means sobre analisis de datos, como afecta su inertia, como su yestimated asigna a un cluster especifico, y como es que un elbow plot nos dice una forma de conocer los clusters que deberiamos tener



```
Print the cluster of image  180 is  2
<Figure size 432x288 with 0 Axes>
```



```
Print the cluster of image  190 is  7
<Figure size 432x288 with 0 Axes>
```



✓ 3s completed at 10:18 AM