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: Pamela Sánchez Arellano
- ID: A01636995
- My carreer: ITC

Importing libraries

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

PART 1

Use your assigned dataset

A1 Load data

```
RunInColab          = True
if RunInColab:
    from google.colab import drive
    drive.mount('/content/drive')

    Ruta            = "/content/drive/My Drive/Colab Notebooks/TC1002S/NotebooksStudents/A01636995"

else:
    Ruta            = "/Users/pamelasanchez/Documents/TC1002S/NotebooksStudents/A01636995/"
```

```
 Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=Tru
```

```
url = Ruta + "/Evidencia/A01636995.csv"

df = pd.read_csv(url)
```

A2 Data managment

Print the first 7 rows

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

	Unnamed: 0	x1	x2
0	0	0.324498	-0.074704
1	1	-0.564642	-1.109772
2	2	0.475341	1.146209
3	3	0.605061	-2.009081
4	4	-0.276323	-0.628257

Print the first 4 last rows

6	6	-0.485153	-1.023259

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

	Unnamed: 0	x1	x2
1196	1196	0.935478	0.222589
1197	1197	-0.002097	1.069751
1198	1198	0.396759	-0.978745
1199	1199	-0.209459	-1.881062

How many rows and columns are in your data?

Use the `shape` method

```
df.shape
```

```
(1200, 3)
```

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

# 1) The rows represent each point, which in turn, has a pair of values assigned to the columns.

# 2) In other words, the columns have two values which represent the dimensional space of each point.
```

Print a statistical summary of your columns

```
df.describe()
```

	Unnamed: 0	x1	x2
count	1200.000000	1200.000000	1200.000000
mean	599.500000	0.252498	-0.500383
std	346.554469	0.513993	0.884925
min	0.000000	-0.816921	-2.340305

```
# 1) What is the minumum and maximum values of each variable
df.x1.max() # 1.366543
df.x1.min() # -0.816921
df.x2.max() # 1.296968
df.x2.min() # -2.340305

# 2) What is the mean and standar deviation of each variable
df.x1.mean() # 0.252498
df.x1.std() # 0.513993
df.x2.mean() # -0.500383
df.x2.std() # 0.884925

# 3) What the 25%, 50% and 75% represent?

# These statistical values represent the distribution of the data by those percentiles.
```

```
0.8849246561762202
```

Rename the columns using the same name with capital letters

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

	Unnamed: 0	X1	X2
0	0	0.324498	-0.074704
1	1	-0.564642	-1.109772
2	2	0.475341	1.146209
3	3	0.605061	-2.009081
4	4	-0.276323	-0.628257
...	...	...	...
1195	1195	0.214794	-0.855080
1196	1196	0.935478	0.222589
1197	1197	-0.002097	1.069751
1198	1198	0.396759	-0.978745
1199	1199	-0.209459	-1.881062

1200 rows × 3 columns

Rename the columns to their original names

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

	Unnamed: 0	x1	x2
0	0	0.324498	-0.074704
1	1	-0.564642	-1.109772
2	2	0.475341	1.146209
3	3	0.605061	-2.009081

Use two different alternatives to get one of the columns

```
df.x1
# or
df.iloc[:,1]
```

```
0       0.324498
1      -0.564642
2       0.475341
3       0.605061
4      -0.276323
          ...
1195    0.214794
1196    0.935478
1197   -0.002097
1198    0.396759
1199   -0.209459
Name: x1, Length: 1200, dtype: float64
```

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

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

	x1	x2
62	0.374003	0.335876
63	-0.392792	-1.555767
64	0.899448	0.238924
65	0.245819	-0.218689
66	0.319313	0.856300
67	-0.171530	-0.085795
68	0.826878	0.587920
69	0.054795	-1.977153
70	0.826154	-0.427434
71	0.835377	-0.016447
72	-0.162674	-0.488868

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

```
df.isnull().sum()
```

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

```
df.notnull().sum()
```

```
Unnamed: 0    1200
x1            1200
x2            1200
dtype: int64
```

Discard the last column (the first*)

```
df.drop("Unnamed: 0", axis=1, inplace = True)
df
```

	x1	x2
0	0.324498	-0.074704
1	-0.564642	-1.109772
2	0.475341	1.146209
3	0.605061	-2.009081
4	-0.276323	-0.628257
...	...	...
1195	0.214794	-0.855080
1196	0.935478	0.222589
1197	-0.002097	1.069751
1198	0.396759	-0.978745
1199	-0.209459	-1.881062

1200 rows × 2 columns

Questions

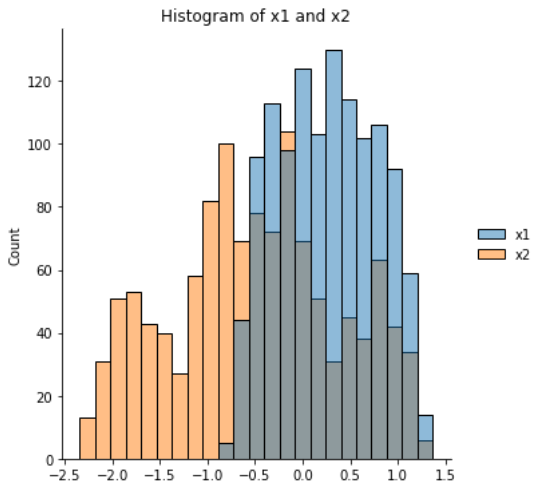
Based on the previos results, provide a description of yout dataset

Your response: My data is composed of 1200 rows and two descrivetive columns. All values are decimal numbers and range from apprx. -2.3 to 1.3. None of the observations are null and all of them are composed of the two variables provided.

A3 Data visualization

Plot in the same figure the histogram of the two variables

```
sns.displot(df, kde = False)
plt.title("Histogram of x1 and x2")
plt.show()
```

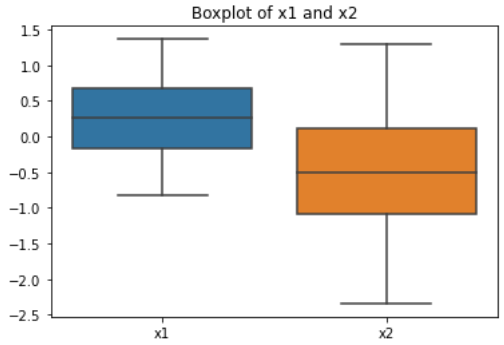


Based on this plots, provide a description of your data:

Your response here: Most of the values are grater than -1.0 and the most repeated values on the count. This means, that most of the points (considering the x1 and x2 values) will be concentraiting in the first quadrant (positive values)

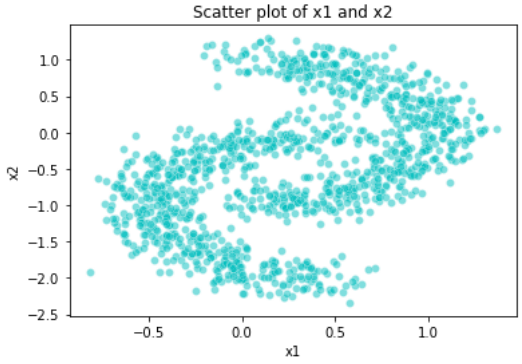
Plot in the same figure the boxplot of the two variables

```
x = df.loc[:, ["x1", "x2"]]
sns.boxplot(data = x)
plt.title("Boxplot of x1 and x2")
plt.show()
```



Scatter plot of the two variables

```
sns.scatterplot(data = df, x = "x1", y = "x2", c = "c", alpha = 0.5)
plt.title("Scatter plot of x1 and x2")
plt.show()
```



Questions

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

Your response: From the boxplot it is seen that the data in x1 is greater than the x2, since the distribution is smaller and shifted to the top of the graph, while the x2 is smaller. Therefore, in the scatter plot the distribution of the data is presented as showned.

A4 Kmeans

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

```
from sklearn.cluster import KMeans

K  =  4
km = KMeans(n_clusters = K, n_init = "auto")
yestimated = km.fit_predict(df)
```

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

```
df.insert(2, "yestimated", yestimated, True)
df.head()
```

	x1	x2	yestimated
0	0.324498	-0.074704	2
1	-0.564642	-1.109772	3
2	0.475341	1.146209	1
3	0.605061	-2.009081	0
4	-0.276323	-0.628257	3

Print the number associated to each cluster

```
df.yestimated.unique()
```

```
array([2, 3, 1, 0], dtype=int32)
```

Print the centroids

```
km.cluster_centers_
```

```
array([[-0.04357085, -1.76433246],
       [ 0.54260007,  0.73738924],
       [ 0.75140603, -0.3037609 ],
       [-0.20589731, -0.69882145]])
```

Print the intertia metric

```
# SSE
km.inertia_
```

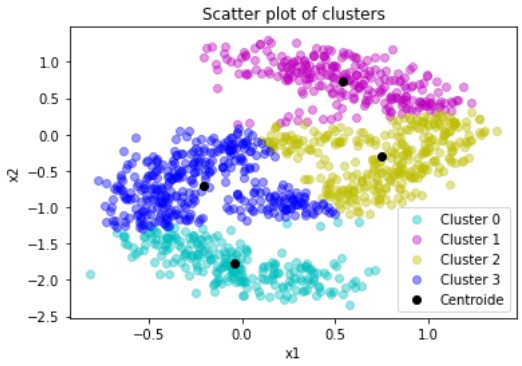
```
236.44392268027448
```

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

```
df1 = df[df.yestimated == 0]
df2 = df[df.yestimated == 1]
df3 = df[df.yestimated == 2]
df4 = df[df.yestimated == 3]

# Scatter plot of each cluster
plt.scatter(df1.x1, df1.x2, label = "Cluster 0", c = 'c', marker = 'o', alpha = 0.4)
plt.scatter(df2.x1, df2.x2, label = "Cluster 1", c = 'm', marker = 'o', alpha = 0.4)
plt.scatter(df3.x1, df3.x2, label = "Cluster 2", c = 'y', marker = 'o', alpha = 0.4)
plt.scatter(df4.x1, df4.x2, label = "Cluster 3", c = 'b', marker = 'o', alpha = 0.4)
# plot centroids
kmc = km.cluster_centers_
plt.scatter(kmc[:,0], kmc[:,1], color="black", label = "Centroide")

plt.title("Scatter plot of clusters")
plt.xlabel("x1")
plt.ylabel("x2")
plt.legend()
plt.show()
```



Questions

Provides a detailed description of your results

Your response: After an exhaustive analysis of the scatterplot, I determined that the data could have 4 clusters. In my examinaton I pictured the four clusters grouped as the 4 cardinal points, which meant that the clusters would have been alligned and more defined. I came to this conclusion since the points overlap at the middle of the plot, which meant that the distribution would be affected if tried another number of clustering.

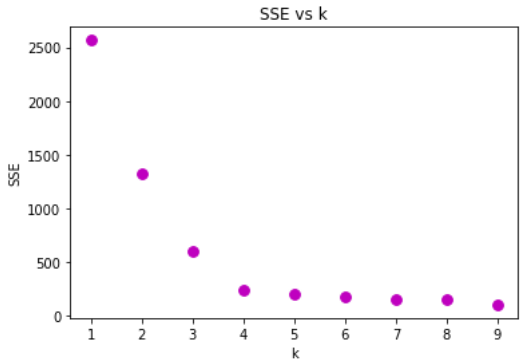
After ploting the scatterplot of the Kmeans clustering, the graph resulted in a not alligned clustering. After analysing the end result of my first examination, I see that the decision to make 4 clusters may not be the best adaptation for the data provided. Therefore, in the next section the elbow plot analysis will be provided to further examinated the data and conclude if my first guess is correct or not.

A5 Elbow plot

Compute the Elbow plot

```
sse = []
kNew = range(1,10)
# For each k
for i in kNew:
  km = KMeans(n_clusters = i, n_init = "auto")
  km.fit_predict(df)
  sse.append(km.inertia_)

plt.scatter(kNew, sse, c = "m", s = 60)
plt.title("SSE vs k")
plt.xlabel("k")
plt.ylabel("SSE")
plt.show()
```



Questions

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

Your response: The best number is 4 since the SSE between the numbers 3 and 4 and their adyecent numbers is the best fit to cluster the data. The SSE greatly decreases from clustering the data with 3 clusters and the difference between 4 and 5 is not noticeable. Therefor the best fit is 4.

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

Your response: Yes. At the begginig, when I saw the data, I noticed the two forms the poits where forming. Nevertheless, I saw the distribution of the data as a 4 group division since the distributions was overlapped in the middle of the plot. This was the turning point to consider 4 clusters. And as seen from the elbow plot analysis, the best fit is also 4 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()
```

```
dict_keys(['data', 'target', 'frame', 'feature_names', 'target_names', 'images', 'DESCR'])
```

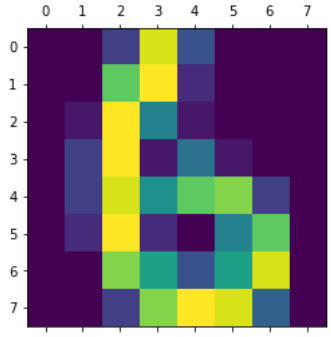
```
# To understand the data
digits.data.shape
```

```
(1797, 64)
```

2) Plot some of the observations

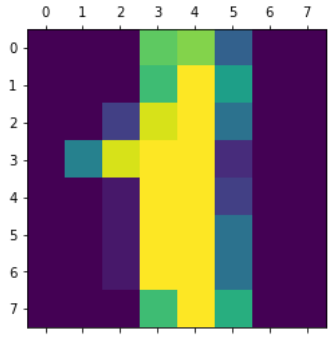
```
plt.matshow(digits.images[234])
```

```
<matplotlib.image.AxesImage at 0x7f59d79bf2b0>
```



```
plt.matshow(digits.images[1])
```

```
<matplotlib.image.AxesImage at 0x7f59da416cd0>
```



3) Do K means clustering

```
# Convert to df
dataframe = pd.DataFrame(digits.data)
dataframe
```

	0	1	2	3	4	5	6	7	8	9	...	54	55	56	57	58	59	60	61	62	63
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	0.0	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	0.0	0.0
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	9.0	0.0
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	0.0	0.0
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	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
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	0.0	0.0
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	0.0	0.0
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	0.0	0.0
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	0.0	0.0
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	1.0	0.0

1797 rows × 64 columns

```
# Kmeans clustering
K  =  10
```

```
km = KMeans(n_clusters = K, n_init = "auto")
yestimated = km.fit_predict(dataframe)
dataframe.insert(64, "yestimated", yestimated, True)
dataframe.head(10)
```

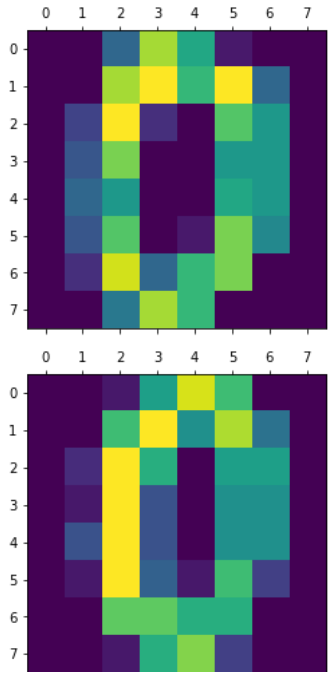
	0	1	2	3	4	5	6	7	8	9	...	55	56	57	58	59	60	61	62	63	yestimated
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	6.0	13.0	10.0	0.0	0.0	0.0	2
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	11.0	16.0	10.0	0.0	0.0	5
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	3.0	11.0	16.0	9.0	0.0	5
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	0.0	0.0	0.0	7.0	13.0	13.0	9.0	0.0	0.0	3
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	2.0	16.0	4.0	0.0	0.0	0
5	0.0	0.0	12.0	10.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	9.0	16.0	16.0	10.0	0.0	0.0	9
6	0.0	0.0	0.0	12.0	13.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	1.0	9.0	15.0	11.0	3.0	0.0	6
7	0.0	0.0	7.0	8.0	13.0	16.0	15.0	1.0	0.0	0.0	...	0.0	0.0	0.0	13.0	5.0	0.0	0.0	0.0	0.0	4
8	0.0	0.0	9.0	14.0	8.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	11.0	16.0	15.0	11.0	1.0	0.0	5
9	0.0	0.0	11.0	12.0	0.0	0.0	0.0	0.0	0.0	2.0	...	0.0	0.0	0.0	9.0	12.0	13.0	3.0	0.0	0.0	9

10 rows × 65 columns

4) Verify your results in any of the observations

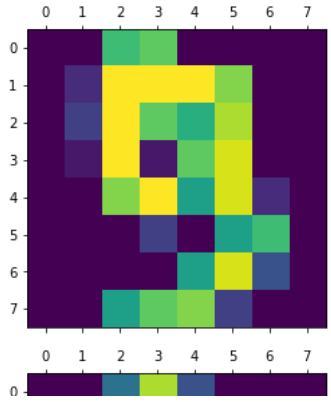
```
# Print multiple images of a digit and then print the cluster it was assigned to
n = [0,10,20,30,49]
for i in n:
  plt.matshow(digits.images[i])
  print(dataframe.yestimated[i])
```

```
2
2
2
2
2
```




```
# Print multiple images of a digit and then print the cluster it was assigned to
n = [9,19,29,39]
for i in n:
  plt.matshow(digits.images[i])
  print(dataframe.yestimated[i])
```

9
9
9
9



Questions

Provides a detailed description of your results.

Your response: To verify the results I used a series of images of the digits 0 and 9, where I could identify without doubt the digit in each selected image. Based on these obsservations, the cluster assigned was able to perfectly separate the digits and assigned a unique cluster to each digit. However, using images where the digit was blur or less defined, the model would make a mistake assigning the cluster. However, it seems like the precision of the model is pretty good and in order to further analyze its accuracy, one would have to make more observations. In general, the model works quite well to identify the digits.

0 1 2 3 4 5 6 7

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: Considero que mi desarrollo de esta subcompetencia es entre sólida, ya que soy capaz de interpretar las variables y su relación con el problema. Además, durante esta semana tec pude desarrollar aún más esta habilidad para la resolución de las actividades diarias y esta última donde esta subcompetencia fue escencial.

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: En esta subcompetencia cosidero mi desarrollo también sólido debido a que soy capaz de construir modelos para explicar y analizar el comportamiento de fenómenos y datos, así como de entender lo que está pasando.

✓ 0s completed at 10:31 AM