GONZALO PAZ

Link Repositorio git@github.com:Cdogonza/InteligenciaArtificial.git

In [ ]:
```
from csv import reader
from math import sqrt
from random import seed
from random import randrange
import pandas as pd
```

In [ ]:
```
def load_csv(filename):
    dataset = list()
    with open(filename, 'r') as file:
        csv_reader = reader(file, delimiter=';')
        for row in csv_reader:
            if not row: # Si la fila está vacía, se salta
                continue
            dataset.append(row)
    return dataset

# Convierte la columna string a float
def str_column_to_float(dataset, column):
    for row in dataset:
        row[column] = float(row[column].strip())

# Encuentra valores min y max para cada columna
def dataset_minmax(dataset):
    minmax = list()
    for i in range(len(dataset[0])):
        col_values = [row[i] for row in dataset] # Crea una lista con los valores de la
        value_min = min(col_values) # Encuentra el valor mínimo
        value_max = max(col_values) # Encuentra el valor máximo
        minmax.append([value_min, value_max]) # Añade los valores a la lista
    return minmax

# Normaliza el dataset entre 0 y 1
def normalize_dataset(dataframe, min_max_values):
    normalized_data = []
    for i, (min_val, max_val) in enumerate(min_max_values):
        col_data = (dataframe.iloc[:, i] - min_val) / (max_val - min_val)
        normalized_data.append(col_data)
    normalized_df = pd.concat(normalized_data, axis=1)
    normalized_df.columns = dataframe.columns
    return normalized_df

# Calcular promedio de columna
def column_means(dataset):
    means = [0 for i in range(len(dataset[0]))] # Crea una lista de 0s con la longitud
    for i in range(len(dataset[0])):
        col_values = [row[i] for row in dataset] # Crea una lista con los valores de la
        means[i] = sum(col_values) / float(len(dataset)) # Calcula el promedio
    return means

# Calcular desviación estándar de columna
def column_stdevs(dataset, means):
    stdevs = [0 for i in range(len(dataset[0]))] # Crea una lista de 0s con la longitud
    for i in range(len(dataset[0])):
        variance = [pow(row[i]-means[i], 2) for row in dataset] # Crea una lista con los
```

```
        stdevs[i] = sum(variance) # Calcula la varianza
    stdevs = [sqrt(x/(float(len(dataset)-1))) for x in stdevs] # Calcula la desviación 
    return stdevs

# Standardize dataset
def standardize_dataset(dataset, mean_values, std_values):
    standardized_data = []
    for i, (mean_val, std_val) in enumerate(zip(mean_values, std_values)):
        col_data = (dataset.iloc[:, i] - mean_val) / std_val
        standardized_data.append(col_data)
    standardized_df = pd.concat(standardized_data, axis=1)
    standardized_df.columns = dataset.columns
    return standardized_df

# Divide el dataset en conjunto de entrenamiento y prueba
def train_test_split(dataset, split=0.60):
    train = list()
    train_size = split * len(dataset) # Calcula el tamaño del conjunto de entrenamiento
    dataset_copy = list(dataset) # Copia el dataset
    while len(train) < train_size: # Mientras el tamaño del conjunto de entrenamiento s
        index = randrange(len(dataset_copy)) # Genera un número aleatorio entre 0 y el 
        train.append(dataset_copy.pop(index)) # Añade el valor del dataset en la posici
    return train, dataset_copy # Devuelve el conjunto de entrenamiento y el conjunto de
```

1. Una vez definidas las funciones que utilizaremos en el programa, procedemos a cargar el dataset y a realizar las operaciones correspondientes.

In [ ]:
```
# Carga del dataset wine.csv proveniente de la misma carpeta
filename = './wine.csv'
column_names = ['class', 'alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesiu
wine = pd.read_csv(filename, names=column_names , sep=';') # Carga el dataset en un dat
print('Número de filas: %d' % len(wine))
```

```
Número de filas: 178
```

1. Imprimir las primeras 10 filas del dataset

In [ ]:
```
# Imprimir las primeras 10 filas del dataset
print(wine.head(10))
```

	class	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium
0	1	1423	1.71	243	156	127
1	1	132	1.78	214	112	100
2	1	1316	2.36	267	186	101
3	1	1437	1.95	25	168	113
4	1	1324	2.59	287	21	118
5	1	142	1.76	245	152	112
6	1	1439	1.87	245	146	96
7	1	1406	2.15	261	176	121
8	1	1483	1.64	217	14	97
9	1	1386	1.35	227	16	98

	total_phenols	flavanoids	Nonflavanoid_phenols	proanthocyanins
0	2.80	3.06	0.28	2.29
1	2.65	2.76	0.26	1.28
2	2.80	3.24	0.30	2.81
3	3.85	3.49	0.24	2.18
4	2.80	2.69	0.39	1.82
5	3.27	3.39	0.34	1.97
6	2.50	2.52	0.30	1.98
7	2.60	2.51	0.31	1.25
8	2.80	2.98	0.29	1.98
9	2.98	3.15	0.22	1.85

	color_intensity	hue	OD280/OD315_of_diluted_wines	proline
0	564	1.04	392	1065
1	438	1.05	34	1050
2	568	1.03	317	1185
3	78	0.86	345	1480
4	432	1.04	293	735
5	675	1.05	285	1450
6	525	1.02	358	1290
7	505	1.06	358	1295
8	52	1.08	285	1045
9	722	1.01	355	1045

1. Convertir strings a float

   No existen valores string en el dataset, por lo que no es necesario realizar esta operación.

2. Obtener los valores min y max de cada columna

In [ ]:
```
# Valores min y max para cada columna del dataset con el nombre de las columnas
minmax = dataset_minmax(wine.values.tolist())
for i in minmax:
    print(wine.columns[minmax.index(i)], i)
```

```
class [1.0, 3.0]
alcohol [12.0, 1483.0]
malic_acid [0.74, 5.8]
ash [2.0, 323.0]
alcalinity_of_ash [12.0, 285.0]
magnesium [70.0, 162.0]
total_phenols [0.98, 3.88]
flavanoids [0.34, 5.08]
Nonflavanoid_phenols [0.13, 0.66]
proanthocyanins [0.41, 3.58]
color_intensity [2.0, 9899999.0]
hue [0.48, 1.71]
OD280/OD315_of_diluted_wines [2.0, 392.0]
proline [278.0, 1680.0]
```

1. Hallar la media de los valores de cada columna

In [ ]:
```
promedio = column_means(wine.values.tolist())
# El promedio de cada columna
for i in promedio:
    print(wine.columns[promedio.index(i)], i)
```

```
class 1.9382022471910112
alcohol 1171.9887640449438
malic_acid 2.336348314606741
ash 190.3370786516854
alcalinity_of_ash 107.47752808988764
magnesium 99.74157303370787
total_phenols 2.295112359550562
flavanoids 2.0292696629213474
Nonflavanoid_phenols 0.36185393258426973
proanthocyanins 1.5908988764044953
color_intensity 55871.7808988764
hue 0.9574494382022468
OD280/OD315_of_diluted_wines 229.97191011235955
proline 746.8932584269663
```

In [ ]:
```
desviacion_std = column_stdevs(wine.values.tolist(), promedio)
# La desviación estándar de cada columna
for i in desviacion_std:
    print(wine.columns[desviacion_std.index(i)], i)
```

```
class 0.7750349899850565
alcohol 374.9340731316293
malic_acid 1.1171460976144627
ash 92.4413931081693
alcalinity_of_ash 89.31896788047699
magnesium 14.282483515295668
total_phenols 0.6258510488339891
flavanoids 0.9988586850169465
Nonflavanoid_phenols 0.12445334029667939
proanthocyanins 0.5723588626747611
color_intensity 742017.2177198853
hue 0.22857156582982338
OD280/OD315_of_diluted_wines 101.65636308491906
proline 314.9074742768489
```

1. Normalizar los valores del dataset

In [ ]:
```
# Normalizar el dataset
wine_norm = normalize_dataset(wine, minmax)
print(wine_norm.head(10))
```

```
   class   alcohol  malic_acid       ash  alcalinity_of_ash  magnesium  \
0    0.0  0.959211    0.191700  0.750779           0.527473   0.619565
1    0.0  0.081577    0.205534  0.660436           0.366300   0.326087
2    0.0  0.886472    0.320158  0.825545           0.637363   0.336957
3    0.0  0.968729    0.239130  0.071651           0.571429   0.467391
4    0.0  0.891910    0.365613  0.887850           0.032967   0.521739
5    0.0  0.088375    0.201581  0.757009           0.512821   0.456522
6    0.0  0.970088    0.223320  0.757009           0.490842   0.282609
7    0.0  0.947655    0.278656  0.806854           0.600733   0.554348
8    0.0  1.000000    0.177866  0.669782           0.007326   0.293478
9    0.0  0.934058    0.120553  0.700935           0.014652   0.304348

   total_phenols  flavanoids  Nonflavanoid_phenols  proanthocyanins  \
0       0.627586    0.573840              0.283019         0.593060
1       0.575862    0.510549              0.245283         0.274448
2       0.627586    0.611814              0.320755         0.757098
3       0.989655    0.664557              0.207547         0.558360
4       0.627586    0.495781              0.490566         0.444795
5       0.789655    0.643460              0.396226         0.492114
6       0.524138    0.459916              0.320755         0.495268
7       0.558621    0.457806              0.339623         0.264984
8       0.627586    0.556962              0.301887         0.495268
9       0.689655    0.592827              0.169811         0.454259

   color_intensity       hue  OD280/OD315_of_diluted_wines   proline
0         0.000057  0.455285                      1.000000  0.561341
1         0.000044  0.463415                      0.082051  0.550642
2         0.000057  0.447154                      0.807692  0.646933
3         0.000008  0.308943                      0.879487  0.857347
4         0.000043  0.455285                      0.746154  0.325963
5         0.000068  0.463415                      0.725641  0.835949
6         0.000053  0.439024                      0.912821  0.721826
7         0.000051  0.471545                      0.912821  0.725392
8         0.000005  0.487805                      0.725641  0.547076
9         0.000073  0.430894                      0.905128  0.547076
```

1. Estandarizar los valores del dataset

In [ ]:
```
wine_std = standardize_dataset(wine, promedio, desviacion_std)
print(wine_std.head(10))
```

```
      class   alcohol  malic_acid       ash  alcalinity_of_ash  magnesium  \
0 -1.210529  0.669481   -0.560668  0.569690           0.543249   1.908522
1 -1.210529 -2.773791   -0.498009  0.255978           0.050633   0.018094
2 -1.210529  0.384097    0.021172  0.829314           0.879124   0.088110
3 -1.210529  0.706821   -0.345835 -1.788561           0.677599   0.928300
4 -1.210529  0.405435    0.227053  1.045667          -0.968188   1.278379
5 -1.210529 -2.747120   -0.515911  0.591325           0.498466   0.858284
6 -1.210529  0.712155   -0.417446  0.591325           0.431291  -0.261969
7 -1.210529  0.624140   -0.166807  0.764408           0.767166   1.488427
8 -1.210529  0.829509   -0.623328  0.288431          -1.046559  -0.191954
9 -1.210529  0.570797   -0.882918  0.396607          -1.024167  -0.121938

   total_phenols  flavanoids  Nonflavanoid_phenols  proanthocyanins  \
0       0.806722    1.031908             -0.657708         1.221438
1       0.567048    0.731565             -0.818411        -0.543189
2       0.806722    1.212114             -0.497005         2.129959
3       2.484437    1.462399             -0.979113         1.029251
4       0.806722    0.661485              0.226158         0.400275
5       1.557699    1.362285             -0.175599         0.662349
6       0.327374    0.491291             -0.497005         0.679820
7       0.487157    0.481280             -0.416654        -0.595603
8       0.806722    0.951817             -0.577356         0.679820
9       1.094330    1.122011             -1.139816         0.452690

   color_intensity       hue  OD280/OD315_of_diluted_wines   proline
0        -0.074537  0.361158                     1.593880  1.010159
1        -0.074707  0.404908                    -1.927788  0.962526
2        -0.074532  0.317409                     0.856101  1.391224
3        -0.075192 -0.426341                     1.131539  2.328007
4        -0.074715  0.361158                     0.620011 -0.037767
5        -0.074387  0.404908                     0.541315  2.232741
6        -0.074590  0.273659                     1.259420  1.724655
7        -0.074617  0.448658                     1.259420  1.740533
8        -0.075227  0.536158                     0.541315  0.946649
9        -0.074324  0.229909                     1.229909  0.946649
```

1. Dividir el dataset en 2 partes: train y test

In [ ]:
```
# Dividir el dataset en conjunto de entrenamiento y prueba
train, test = train_test_split(wine.values.tolist(), 0.60)
print('Número de filas de entrenamiento: %d' % len(train))
print('Número de filas de prueba: %d' % len(test))
```

```
Número de filas de entrenamiento: 107
Número de filas de prueba: 71
```

In [ ]:
```
# Imprimir las primeras 10 filas del conjunto de entrenamiento, con el nombre de cada c
train = pd.DataFrame(train, columns=column_names)
print(train.head(10))
```

```
   class  alcohol  malic_acid    ash  alcalinity_of_ash  magnesium  \
0    2.0   1221.0        1.19  175.0              168.0      151.0
1    3.0   1225.0        3.88   22.0              185.0      112.0
2    2.0   1264.0        1.36  202.0              168.0      100.0
3    1.0   1376.0        1.53   27.0              195.0      132.0
4    1.0   1422.0        1.70   23.0              163.0      118.0
5    1.0    141.0        2.02   24.0              188.0      103.0
6    2.0   1334.0        0.94  236.0               17.0      110.0
7    1.0   1305.0        2.05  322.0               25.0      124.0
8    2.0   1165.0        1.67  262.0               26.0       88.0
9    1.0   1285.0        1.60  252.0              178.0       95.0

   total_phenols  flavanoids  Nonflavanoid_phenols  proanthocyanins  \
0           1.85        1.28                  0.14             2.50
1           1.38        0.78                  0.29             1.14
2           2.02        1.41                  0.53             0.62
3           2.95        2.74                  0.50             1.35
4           3.20        3.00                  0.26             2.03
5           2.75        2.92                  0.32             2.38
6           2.53        1.30                  0.55             0.42
7           2.63        2.68                  0.47             1.92
8           1.92        1.61                  0.40             1.34
9           2.48        2.37                  0.26             1.46

   color_intensity   hue  OD280/OD315_of_diluted_wines  proline
0            285.0  1.28                         307.0    718.0
1            821.0  0.65                           2.0    855.0
2            575.0  0.98                         159.0    450.0
3             54.0  1.25                           3.0   1235.0
4            638.0  0.94                         331.0    970.0
5             62.0  1.07                         275.0   1060.0
6            317.0  1.02                         193.0    750.0
7            358.0  1.13                          32.0    830.0
8             26.0  1.36                         321.0    562.0
9            393.0  1.09                         363.0   1015.0
```

In [ ]:
```
# Imprimir las primeras 10 filas del conjunto de prueba
test= pd.DataFrame(test, columns=column_names)
print(test.head(10))
```

	class	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium
0	1.0	132.0	1.78	214.0	112.0	100.0
1	1.0	1324.0	2.59	287.0	21.0	118.0
2	1.0	142.0	1.76	245.0	152.0	112.0
3	1.0	1406.0	2.15	261.0	176.0	121.0
4	1.0	1483.0	1.64	217.0	14.0	97.0
5	1.0	141.0	2.16	23.0	18.0	105.0
6	1.0	1375.0	1.73	241.0	16.0	89.0
7	1.0	1475.0	1.73	239.0	114.0	91.0
8	1.0	1438.0	1.87	238.0	12.0	102.0
9	1.0	1363.0	1.81	27.0	172.0	112.0

	total_phenols	flavanoids	Nonflavanoid_phenols	proanthocyanins
0	2.65	2.76	0.26	1.28
1	2.80	2.69	0.39	1.82
2	3.27	3.39	0.34	1.97
3	2.60	2.51	0.31	1.25
4	2.80	2.98	0.29	1.98
5	2.95	3.32	0.22	2.38
6	2.60	2.76	0.29	1.81
7	3.10	3.69	0.43	2.81
8	3.30	3.64	0.29	2.96
9	2.85	2.91	0.30	1.46

	color_intensity	hue	OD280/OD315_of_diluted_wines	proline
0	438.0	1.05	34.0	1050.0
1	432.0	1.04	293.0	735.0
2	675.0	1.05	285.0	1450.0
3	505.0	1.06	358.0	1295.0
4	52.0	1.08	285.0	1045.0
5	575.0	1.25	317.0	1510.0
6	56.0	1.15	29.0	1320.0
7	54.0	1.25	273.0	1150.0
8	75.0	1.20	3.0	1547.0
9	73.0	1.28	288.0	1310.0