Análisis de ciencia de datos (TC-2004B)

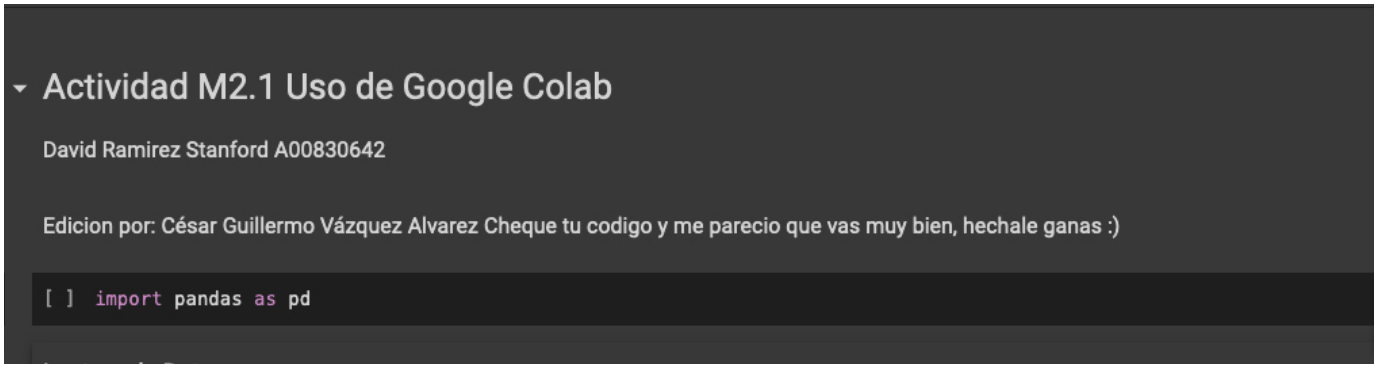
César Guillermo Vázquez Alvarez A01197857
IDM - Ingeniería de Ciencia de Datos y Matemáticas

Actividad M2.1 Uso de Google Colab

Aportación de compañero:

```
#Edicion David Ramirez Stanford A00830642
```

Aportación mia



Lectrura de datos

```
# Importar librería Pandas
import pandas as pd
# Acceder a datos en Google Drive
from google.colab import drive
drive.mount('/content/drive')
```

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

```
# Se lee el archivo CSV desde google drive,
df = pd.read_csv('/content/drive/MyDrive/CienciaDeDatos/titanic.csv')
```

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tick
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 211
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th	female	38.0	1	0	PC 175

```
# Se muestran las últimas filas del DataFrame
df.tail()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tick
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	2115
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	1120
				Johnston					

Descripción de variables.

```
# Muestra el tipo de dato en cada columna.
df.dtypes
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

La información es tipo categorica o numérica:

1. PassengerId: Numérica 2. Survived: Numérica
2. PClass: Numérica
3. Name: Categórica
4. Sex: Categórica
5. Age: Numérica
6. SibSp: Numérica 8. Parch: Numérica 9. Ticket: Categórica
7. Fare: Numérica
8. Cabin: Categórica
9. Embarked: Categórica

```
# Muestra el nombre de las etiquetas de las columnas.
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

```
# Imprime un resumen del data frame.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
# Muestra los datos estadisticos del DataFrame.
```

```
df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	89
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	3
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	4
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	1
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	3
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	51

¿Cuántos registros se tienen en total?

```
print('Fueron', len(df), 'pasajeros')
```

```
Fueron 891 pasajeros
```

¿Cuáles tienen valores nulos?

```
# Se suma la cantidad de valores nulos.
df.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

¿Cuáles no tienen valores nulos?

```
# Se suma la cantidad de valores no nulos.
df.notnull().sum()
```

```
PassengerId    891
Survived       891
Pclass         891
Name           891
Sex            891
Age            714
SibSp          891
Parch          891
Ticket         891
Fare           891
Cabin          204
Embarked       889
dtype: int64
```

¿Qué columnas aparecen en el resultado de describe?

```
# Solo se muestran los datos que no tienen valores nulos.
df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	89
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	3
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	4
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	1
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	3
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	51

¿Cuál es la tarifa más cara?

```
print('La tarifa más cara fue de: $', df['Fare'].max())
```

```
La tarifa más cara fue de: $ 512.3292
```

La tarifa más cara fue de: $ 512.3292

¿Cuál es el promedio de edad?

```
print('Promedio de edad:', df['Age'].mean())
```

```
Promedio de edad: 29.69911764705882
```

En promedio la edad de los pasajeros fue de 30 años.

¿Cuántos valores diferentes se tiene por categoría?

```
nombre = df['Name'].unique()
print('De la columna Name hay', len(nombre), 'valores diferentes.')
print('\n')
nombre
```

```
De la columna Name hay 891 valores diferentes.


array(['Braund, Mr. Owen Harris',
       'Cumings, Mrs. John Bradley (Florence Briggs Thayer)',
       'Heikkinen, Miss. Laina',
       'Futrelle, Mrs. Jacques Heath (Lily May Peel)',
       'Allen, Mr. William Henry', 'Moran, Mr. James',
       'McCarthy, Mr. Timothy J', 'Palsson, Master. Gosta Leonard',
       'Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)',
       'Nasser, Mrs. Nicholas (Adele Achem)',
       'Sandstrom, Miss. Marguerite Rut', 'Bonnell, Miss. Elizabeth',
       'Saundercock, Mr. William Henry', 'Andersson, Mr. Anders Johan',
       'Vestrom, Miss. Hulda Amanda Adolfina',
       'Hewlett, Mrs. (Mary D Kingcome) ', 'Rice, Master. Eugene',
       'Williams, Mr. Charles Eugene',
       'Vander Planke, Mrs. Julius (Emelia Maria Vandemoortele)',
       'Masselmani, Mrs. Fatima', 'Fynney, Mr. Joseph J',
       'Beesley, Mr. Lawrence', 'McGowan, Miss. Anna "Annie"',
       'Sloper, Mr. William Thompson', 'Palsson, Miss. Torborg Danira',
       'Asplund, Mrs. Carl Oscar (Selma Augusta Emilia Johansson)',
       'Emir, Mr. Farred Chehab', 'Fortune, Mr. Charles Alexander',
       'O\'Dwyer, Miss. Ellen "Nellie"', 'Todoroff, Mr. Lalio',
       'Uruchurtu, Don. Manuel E',
       'Spencer, Mrs. William Augustus (Marie Eugenie)',
       'Glynn, Miss. Mary Agatha', 'Wheadon, Mr. Edward H',
       'Meyer, Mr. Edgar Joseph', 'Holverson, Mr. Alexander Oskar',
       'Mamee, Mr. Hanna', 'Cann, Mr. Ernest Charles',
       'Vander Planke, Miss. Augusta Maria',
       'Nicola-Yarred, Miss. Jamila',
       'Ahlin, Mrs. Johan (Johanna Persdotter Larsson)',
```

```
'Antin, Mrs. Jonan (Johanna Persdotter Larsson)',
'Turpin, Mrs. William John Robert (Dorothy Ann Wonnacott)',
'Kraeff, Mr. Theodor', 'Laroche, Miss. Simonne Marie Anne Andree',
'Devaney, Miss. Margaret Delia', 'Rogers, Mr. William John',
'Lennon, Mr. Denis', "O'Driscoll, Miss. Bridget",
'Samaan, Mr. Youssef',
'Arnold-Franchi, Mrs. Josef (Josefine Franchi)',
'Panula, Master. Juha Niilo', 'Nosworthy, Mr. Richard Cater',
'Harper, Mrs. Henry Sleeper (Myna Haxtun)',
'Faunthorpe, Mrs. Lizzie (Elizabeth Anne Wilkinson)',
'Ostby, Mr. Engelhart Cornelius', 'Woolner, Mr. Hugh',
'Rugg, Miss. Emily', 'Novel, Mr. Mansouer',
'West, Miss. Constance Mirium',
'Goodwin, Master. William Frederick', 'Sirayanian, Mr. Orsen',
'Icard, Miss. Amelie', 'Harris, Mr. Henry Birkhardt',
'Skoog, Master. Harald', 'Stewart, Mr. Albert A',
'Moubarek, Master. Gerios', 'Nye, Mrs. (Elizabeth Ramell)',
'Crease, Mr. Ernest James', 'Andersson, Miss. Erna Alexandra',
'Kink, Mr. Vincenz', 'Jenkin, Mr. Stephen Curnow',
'Goodwin, Miss. Lillian Amy', 'Hood, Mr. Ambrose Jr',
'Chronopoulos, Mr. Apostolos', 'Bing, Mr. Lee',
'Moen, Mr. Sigurd Hansen', 'Staneff, Mr. Ivan',
'Moutal, Mr. Rahamin Haim', 'Caldwell, Master. Alden Gates',
'Dowdell, Miss. Elizabeth', 'Waelens, Mr. Achille',
'Sheerlinck, Mr. Jan Baptist', 'McDermott, Miss. Brigdet Delia',
'Carrau, Mr. Francisco M', 'Ilett, Miss. Bertha',
'Backstrom, Mrs. Karl Alfred (Maria Mathilda Gustafsson)',
'Ford, Mr. William Neal', 'Slocovski, Mr. Selman Francis',
```

```
for i in list(df.columns):
  categories = df[i].unique()
  print('De la columna', i,'hay', len(categories), 'valores diferentes.')
  display(categories)
  print("\n\n\n----------------------------------------------------------------
```

```
De la columna PassengerId hay 891 valores diferentes.
array([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,  13,
        14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,  26,
        27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,  39,
        40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,  52,
        53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,  65,
        66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,  78,
        79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,  91,
        92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103, 104,
       105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117,
       118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130,
       131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143,
       144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156,
       157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169,
       170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182,
```

```
      183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195,
      196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208,
      209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221,
      222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234,
      235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247,
      248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260,
      261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273,
      274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286,
      287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299,
      300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312,
      313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325,
      326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338,
      339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351,
      352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364,
      365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377,
      378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390,
      391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403,
      404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416,
      417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429,
      430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442,
      443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455,
      456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468,
      469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481,
      482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494,
      495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507,
      508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520,
      521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533,
      534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546,
      547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559,
      560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572,
      573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585,
      586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598,
      599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611,
      612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624,
      625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637,
      638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650,
      651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663,
      664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676,
      677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689,
```

Análisis de Datos

¿Cuánto es el total que se pagó sumando la tarifa de todos los pasajeros?

```
print('El total que se pagó fue: $', df['Fare'].sum())
```

```
El total que se pagó fue: $ 28693.9493
```

Subconjunto de los datos de los que sobrevivieron:

```
sobrevivieron = df[df['Survived'] == 1]
sobrevivieron
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ti
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 1
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STO 310
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	11
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	34

Subconjunto de los datos de los que no sobrevivieron:

```
noSobrevivieron = df[df['Survived'] == 0]
noSobrevivieron
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	T
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	3
5	6	0	3	Moran, Mr. James	male	NaN	0	0	3
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	3
...	...	...	...	...	...	...	...	...	

Datos de 5 personas que sobrevivieron:

```
df1 = df[df['Survived'] == 1]
df1.head(5)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tick
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 175
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O 31012

Datos de 5 personas que no sobrevivieron:

```
df1 = df[df['Survived'] == 0]
df1.head(5)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450
5	6	0	3	Moran, Mr.	male	NaN	0	0	330877