**Universidad CAECE**



**Taller de Lenguajes II**

Prof. Silvia A. Cobialca

**Informe  
‘UN: World Happiness Report’**

Andrés Garcia Alves de Borba  
leg. 1033744

**TABLA DE CONTENIDOS**

[Consigna 4](#__RefHeading___Toc683_3764710455)

[Consigna #1: Desarrollo 5](#__RefHeading___Toc824_863528844)

[→ Intro 5](#__RefHeading___Toc826_863528844)

[→ Hipótesis 5](#__RefHeading___Toc820_863528844)

[→ Exploración Inicial 6](#__RefHeading___Toc822_863528844)

[→ Generalizaciones 8](#__RefHeading___Toc812_863528844)

[→ Distribución Poblacional 9](#__RefHeading___Toc1024_3686425629)

[→ Valores Atípicos 13](#__RefHeading___Toc1030_3686425629)

[→ Correlaciones 16](#__RefHeading___Toc1169_4094841946)

[→ Regresión Lineal 20](#__RefHeading___Toc1072_3644636583)

[→ Validación del Modelo 24](#__RefHeading___Toc1222_1394717092)

[Consigna #2: Desarrollo 27](#__RefHeading___Toc1229_1394717092)

[→ Intro 27](#__RefHeading___Toc1225_1394717092)

[→ Hipótesis 27](#__RefHeading___Toc1429_492608335)

[→ Distribución Poblacional (TODO) 28](#__RefHeading___Toc1435_492608335)

[Anexo I: Código Fuente 29](#__RefHeading___Toc742_2257247263)

[Anexo II: Librería MATPLOTLIB 30](#__RefHeading___Toc3007_515470991)

[→ Gráficos de Funciones 32](#__RefHeading___Toc319_1001808647)

[→ Gráficos tipo Torta 34](#__RefHeading___Toc363_1341352657)

[→ Gráficos tipo Histograma 36](#__RefHeading___Toc830_863528844)

[→ Diagramas de Dispersión (Scatter) 37](#__RefHeading___Toc365_1341352657)

[→ Diagramas de Caja (Boxplot) 38](#__RefHeading___Toc3120_585838588)

[Anexo III: Librería SEABORN 39](#__RefHeading___Toc3015_515470991)

[→ Gráficos de Barras 40](#__RefHeading___Toc310_315084060)

[→ Gráficos de Pares 42](#__RefHeading___Toc3043_515470991)

[→ Gráficos tipo Joint 43](#__RefHeading___Toc3017_515470991)

[→ Gráficos tipo KDE 45](#__RefHeading___Toc3025_515470991)

[→ Gráficos tipo LM 46](#__RefHeading___Toc3045_515470991)

[→ Gráficos tipo Violín 47](#__RefHeading___Toc3021_515470991)

[Referencias 49](#__RefHeading___Toc439_3024620491)

# Consigna

#1 - 2020.05.28  
  
Partiendo desde datos extraidos del [Reporte Mundial de Felicidad](https://worldhappiness.report/), publicados por la ONU [1] [2], se plantea el objetivo de medir y/o comparar los factores que influyen en la felicidad de cada país.

Durante el proceso, se buscará mostrar las métricas e interpretar los gráficos, con un mayor énfasis en la interpretacion.

#2 - 2020.06.04  
  
Enriquecer el dataframe original del Reporte Mundial de Felicidad con datos complementarios.

Algunas opciones:

a) Indices de criminalidad de la ONU [3][4].

b) Algún otro índice que consideremos oportuno.

# Consigna #1: Desarrollo

## → Intro

Como parte del desarrollo del presente informe, me propongo empezar por un análisis de correlaciones sobre los datos, luego una regresión lineal, para finalmente terminar esta etapa con la correspondientes validaciones sobre los resultados.

En el camino iré mostrando las métricas correspondientes, y seleccionando gráficos y sus explicaciones que permitan una mejor comprensión de los datos.

## → Hipótesis

Me planteo como hipótesis que la variable Felicidad sea en gral la que mayor impacto tenga sobre la felicidad de las personas.

## → Exploración Inicial

Como primer paso a cualquier análisis, realizo una exploración inicial de la estructura de los datos a analizar. Esto implica familiarizarme con la cantidad de observaciones disponibles, las variables de estas, sus tipos de datos, entre otros.

|  |
| --- |
|  |
| Estructura de datos del dataset. |

Se puede ver aquí que el dataframe está compuesto por 158 observaciones, con 12 variables cada una.

- Las primeras 2 variables ‘Pais’ y ‘Región’ con netamente descriptivas.  
- Las siguientes 3 ‘Ranking’, ‘Puntaje’, ‘ErrorStd’ del tipo totalizadoras.  
- Y finalmente a efectos de analizar sus impactos sobre la variable PuntajeFelicidad, se pueden utilizar las últimas 7: Economia, Familia, ExpectVida, Libertad, Instituciones, Generosidad, Justicia.

|  |
| --- |
|  |
| Muestra de sus datos. |

|  |
| --- |
|  |
| Subregiones incluidas. |

## → Generalizaciones

Dado que el dataframe original cuenta con una variable dedicada al mostar el error estandar del índice de felicidad para cada país, aprovecho este para medir en promedio su error absoluto y su error relativo, y con esto hacerme una idea de la calidad de los datos incluidos.

|  |
| --- |
|  |
| Error relativo = (Error absoluto / Valor original) \* 100 |

Resultando en el promedio de las observaciones un error relativo menor al 1%.

## → Distribución Poblacional

A continuación chequeo algunas variables del dataframe mediante histogramas. Con estos se puede visualizar un panorama general de la distribución poblacional para la variable dada.

|  |
| --- |
|  |
| Código fuente. |

|  |
| --- |
|  |
| Países por región. |

Siendo la muestra de 1 observación por país, es esperable diferentes tamaños poblacionales en las distintas subregiones del mundo.  
  
En un extremo se puede observar a norteamérica y oceanía con 3 ocurrencias cada una, y por otro al áfrica subsahariana con casi 40.

|  |
| --- |
|  |
| Puntuaciones en Felicidad. |

Sobre el índice de felicidad ya se empieza vislumbrar una pseudo forma acampanada, típica de las distribuciones normales, con centro entre los valores 4 a 6.

|  |
| --- |
|  |

Donde para este rango en efecto se verifica una cantidad de ocurrencias cercana al 60%, característica del Sigma-1 (68%) en una distribución normal.

|  |
| --- |
|  |
| Puntuaciones en economía. |

La variable elegida a analizar como más influyente sobre el índice de felicidad, también muestra una gran concentración de ocurrencias para los valores centrales, aunque en este caso algo desbalanceada hacia la izquierda.

## **→ Valores Atípicos**

Los valores atípicos (outliers) son observaciones numéricamente distantes del resto de los datos.  
Como paso previo al proceso de análisis de variables, sería conveniente descartarlos, ya que estos pueden distorcionar las predicciones y afectar la precisión.

Como en este dataframe existe una única observación consolidada por pais, el enfoque más simple de eliminar la observación no resulta viable. No obstante lo cual realizo igual la búsqueda de outliers, para llegado buscar métodos paliativos.

|  |
| --- |
| dsBoxPlot = pd.DataFrame(ds2015, columns=['PuntajeFelicidad','Economia','Familia',  'ExpectVida','Libertad','Instituciones','Generosidad','Justicia'])  # normalizar los datos (rango de 0 a 1)  dsBoxPlot.PuntajeFelicidad = dsBoxPlot.PuntajeFelicidad /  max(dsBoxPlot.PuntajeFelicidad)  dsBoxPlot.Economia = dsBoxPlot.Economia / max(dsBoxPlot.Economia)  dsBoxPlot.Familia = dsBoxPlot.Familia / max(dsBoxPlot.Familia)  dsBoxPlot.ExpectVida = dsBoxPlot.ExpectVida / max(dsBoxPlot.ExpectVida)  dsBoxPlot.Libertad = dsBoxPlot.Libertad / max(dsBoxPlot.Libertad)  dsBoxPlot.Instituciones = dsBoxPlot.Instituciones / max(dsBoxPlot.Instituciones)  dsBoxPlot.Generosidad = dsBoxPlot.Generosidad / max(dsBoxPlot.Generosidad)  dsBoxPlot.Justicia = dsBoxPlot.Justicia / max(dsBoxPlot.Justicia)  plt.figure(figsize=(10,6))  labels = ['Felicidad','Economia','Familia','ExpectVida','Libertad',  'Instituciones','Generosidad','Justicia']  # diagrama de cajas  dsBoxPlot = [dsBoxPlot.PuntajeFelicidad, dsBoxPlot.Economia, dsBoxPlot.Familia,  dsBoxPlot.ExpectVida, dsBoxPlot.Libertad, dsBoxPlot.Instituciones,  dsBoxPlot.Generosidad, dsBoxPlot.Justicia]  plt.boxplot(dsBoxPlot, labels=labels)  plt.show() |

|  |
| --- |
|  |
| Diagrama de cajas, utilizado en la detección de outliers. |

Me sorprende que siendo datos provenientes de la ONU, encontrar una cantidad tan grotesca de valores atípicos para la variable Instituciones, y que estos no hayan sido suprimidos previo a su consolidación por país.  
  
Como medida paliativa, una opción será descartar esta variable de los futuros análisis.

La situación me da pie para pobar también el tipo de gráfico violín: estos muestran la distribución cuantitativa de los datos a través de varios niveles (al igual que bloxplot), permitiendo que estas distribuciones puedan compararse.

|  |
| --- |
|  |
| Código fuente. |

|  |  |
| --- | --- |
|  |  |
| Comparativa de 2 variables mediante gráfico tipo violín. | |

En cada una de sus secciones por el **espesor de la figura** se puede comparar visualmente la cantidad de observaciones que abarca.  
  
De la gráfica de la variable Instituciones, se visualiza que la mayoría de los valores caen en el rango aprox de 0 – 0,2 y que se presentan algunos outliers a partir del valor 0,5.  
  
Por constraste, en la gráfica de al lado sobre la variable Economia, puede visualizarse que no presenta valores atípicos, y que los valores se hayan distribuidas de forma mucho más pareja a lo largo del rango en cuestión.

## → Correlaciones

A continuación paso a plotear algunas variables a través de los diagramas de pares y de dispersión, que me puedan sugerir correlaciones entre ellas...

|  |
| --- |
|  |
| Diagragrama de pares. |

El uso del diagrama PairPlot de la librería Seaborn, permite realizar un análisis visual de todas las posibles correlaciones entre las distintas variables del dataframe dado.  
  
De las gráficas, observo una clara tendencia entre **Felicidad** con las 3 primeras columnas: **Economia**, **Familia** y **Expectativa de Vida**, donde a mayor valor en cada una de estas (variables independientes), mayor el valor resultante en Felicidad (variable dependiente).

Momento propicio para realizar un diagrama de dispersión y visualizar lo mismo que en el diagrama de pares, pero para dos variables en concreto:

|  |
| --- |
|  |
|  |
| Diagrama de dispersión: la felicidad en relación a la economía. |

Claramente en este último diagrama se puede observar que dado un incremento para el índice económico existe una asociación con el incremento de la felicidad.  
  
Sabiendo estas posibles correlaciones, ahora paso al cálculo de las métricas correspondientes:

|  |
| --- |
|  |
| Coeficientes de correlación |

En efecto, el cálculo de los coeficientes muestra para los 3 casos correlaciones fuertes con la variable Felicidad, con índices mayores al 70%.

## **→ Regresión Lineal**

En los pasos anteriores se ha identificado que la variable Felicidad posee cierta colinealidad con las variables Economía, Familia y Expectativa de Vida.  
  
Para identificar con que factores intervienen estas variables en la felicidad de los paises, paso a realizar una regresión lineal sobre los datos.

|  |
| --- |
|  |
| Preparación de los datos. |

Una salvedad:  
Dada la versatilidad de la librería StatsModels [5], es que quiero aprovechar para realizar las métricas de esta parte del informe mediante una Regresión Lineal Múltiple, sobre las variables Economia, Familia y Expectativa de Vida, en su conjunto.  
Aunque para las visualizaciones posteriores en las librerías Matplotlib, Seaborn, YellowBrick y otros módulos de la propia StatsModels, hasta donde conozco, solo soportan la graficación de modelos de Regresión Lineal Simple.

|  |
| --- |
|  |
|  |
| Regresión lineal |

De las métricas cabe destacar:

- La columna **coef**, que devuelve el peso calculado para cada variable.  
- La columan **Pr(>|t|)**, que muestra el p-value (significancia) en las variables.  
  
En el modelo generado, tanto la ordenada al origen (Intercept) como las variables de la pendiente (x1, x2, x3) son significativas ya que sus p-values son todos menores a 0.05.

|  |
| --- |
|  |
| La ecuación de la recta. |

Llegado este punto ya se puede hacer uso de los datos del dataframe original, que se reservaron de forma separada para contrastar el modelo…

|  |
| --- |
|  |

A continuación paso a mostrar:

→ Que tanto el dataframe de Train, como el de Test, han sido constuidos con datos tomados de forma aleatoria desde el dataframe original.

→ Que la variable dependiente guarda semejanza, entre los valores de test originales, y los valores predichos por del modelo.

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

## **→ Validación del Modelo**

Para poder confiar en un modelo de regresión lineal es necesario que se cumpla con los siguientes supuestos:

→ Linealidad: Que la relación entre las variables sea lineal.  
 → Que los errores en la mediciones de las variables (residuos) sean independientes entre sí.  
 → Homocedasticidad: Que los errores tengan varianza constante.  
 → Distribución normal de los residuos: Que los errores tengan una esperanza matemática igual a cero.  
 → Que el error total sea la suma de todos los errores.  
 → Se debe estudiar con detenimiento los valores atípicos ya que pueden generar una falsa correlación que realmente no existe, u ocultar una existente.

Residuos = Valores observados - Valores que predice el modelo (e = y - ŷ)

Para validar la homocedasticidad y la linealidad del modelo se suele utilizar un gráfico de residuos.  
  
En este, mediante una inspección visual se debe verificar la aleatoriedad de los mismos:

|  |
| --- |
|  |
| Aleatoriedad de los residuos, tanto en el conjunto de train como el de test. |

Luego, para validar la hipótesis de normalidad se suele realizar mediante un gráfico QQ-Normal, también sobre los residuos. Este gráfico mostrará una comparación cuartil a cuartil de los datos recibidos contra una distribución normal teórica.  
  
De verificarse, se debe observar que los residuos siguen aproximadamente la línea recta diagonal.

|  |
| --- |
|  |
| QQ-plot: Los residuos como puntos azules y en la línea roja la distribución ideal. |

# Consigna #2: Desarrollo

## **→ Intro**

En este capítulo buscaré cumplimentar la 2da consigna: enriquecer el análisis del ‘Indice de Felicidad’ contra algún otro indicador externo. Para ello he elegido utilizar el ‘Indice de Alfabetización’ por países.  
  
A los efectos prácticos de este informe, tomaré los datos provistos por [DatosMacro.com](https://datosmacro.expansion.com/demografia/tasa-alfabetizacion) [6].  
Luego, para los casos faltantes, completando a mano desde [Wikipedia.org](https://es.wikipedia.org/wiki/Anexo:Países_por_tasa_de_alfabetización) [7] (Informe UNESCO 2010) y desde [CIA: The World Factbook](https://www.cia.gov/library/publications/the-world-factbook/fields/370.html) [8].

Me propongo determinar una posible **asociación** entre la **Educación** y la **Felicidad** de las personas.  
  
Reconozco que el Indice de Alfabetización de un país, puede resultar algo básico a este efecto, ya que no distingue es ese caso el nivel académico alcanzado por la población en general.  
  
Por otro lado, un índice que midiera este aspecto en forma directa sería una mejora, pero no perfecto, dadas las diferencias entre países para los contenidos académicos, el nivel de las instituciones, y demaces aspectos educativos aún entre títulos educativos del mismo nivel.  
Luego pensé en el conocidísimo Informe PISA, pero este mide calidad educativa, no así nivel (grado) educativo alcanzado por la población, con el agravante que cubre menos del 50% de los paises del mundo.  
  
Pese a sus limitaciones, considero que el Indice de Alfabetización puede ser una primera aproximación válida al estudio de asociaciones entre Educación y Felicidad.

## → Hipótesis

Que la Educación y la Felicidad se encuentran altamente asociadas, observable mediante un factor de correlación de almenos 0,70.

## → Distribución Poblacional (TODO)

Seguro queda desbalanceado (la mayoria tiene > 80%): filtrar N/A y outliers  
TODO !!!

# Anexo I: Código Fuente

El presente informe, el código fuente utilizado en los scripts, los dataframes, las capturas de pantalla, y otros recursos varios, están disponibles para su consulta en el siguiente GitHub:  
  
<https://github.com/andres-garcia-alves/world-happiness-report>

# Anexo II: Librería MATPLOTLIB

Matplotlib es una completa biblioteca open-source para crear visualizaciones estáticas, animadas e interactivas en Python.

Sitio oficial: [https://matplotlib.org](https://matplotlib.org/)  
GitHub oficial: <https://github.com/matplotlib/matplotlib>

Se puede instalar mediante el comando:

pip install matplotlib

Esta posee a su vez las siguientes dependencias:

python 3.6+  
numpy 1.11+  
freetype 2.3+  
libpng 1.2+  
cycler 0.10+  
dateutil 2.1+  
wikisolver 1.0+

Se importa de la siguiente manera:

import matplotlib.pyplot as plt # visualización de datos

Se desarrolla en las próximas secciones un análisis de algunos de sus tipos de gráficos incluidos.

|  |
| --- |
|  |
| La anatomía de un gráfico en matplotlib. |

## → Gráficos de Funciones

matplotlib.pyplot.plot()  
  
  
El gráfico más simple que se puede constuir en esta librería es la **representación de una función** desde una operación matemática o una secuencia de puntos.

Se representa en una línea contínua la unión de los puntos de la variable dependiente (eje Y) asociados a sus equivalente desde la variable independiente (eje X).

Todas las funciones de trazado esperan numpy.array o numpy.ma.masked\_array como entrada. Las clases que son "tipo matriz", como los objetos de datos pandas y numpy.matrix, pueden o no funcionar según lo previsto. Es mejor convertirlos en objetos numpy.array antes de graficar. [9]

**Ejemplo**

plt.title('Felicidad por Region') # título al gráfico

plt.xticks(rotation=90) # etiquetas del eje x verticales

# plt.xlabel("Region") # etiqueta eje x (var independiente)

plt.ylabel('Felicidad') # etiqueta eje y (var dependiente)

# gráfico de la función

plt.plot(dsFelicidad.region, dsFelicidad.puntaje)

plt.show()

|  |
| --- |
|  |
| Muestra de los datos para el gráfico. |

|  |
| --- |
|  |
| Ejemplo de matplotlib.pyplot.plot() |

## → Gráficos tipo Torta

matplotlib.pyplot.pie()  
  
  
Siguendo en una posible calificación por grado de complejidad, se encontrarían los gráficos de torta. Este es un recurso [estadístico](https://es.wikipedia.org/wiki/Estadística) que se utiliza para **representar** [**porcentajes**](https://es.wikipedia.org/wiki/Porcentaje) **y** [**proporciones**](https://es.wikipedia.org/wiki/Proporción). El número de elementos comparados dentro de una gráfica circular suele ser de más de cuatro.

Se utilizan en aquellos casos donde interesa no solamente mostrar el número de veces que se dan una característica o atributo de manera tabular sino más bien de manera gráfica, de tal manera que se pueda visualizar mejor la proporción en que aparece esa característica respecto del total. [10]

El método pie() realiza un gráfico circular desde el array x provisto. El área fraccional de cada porción viene dada por la fórmula x / sum(x). If sum(x) < 1, entonces los valores de x dan directamente el área fraccional y la matriz no se normalizará. El gráfico de torta resultante tendrá una porción vacía de tamaño 1 - suma(x). Las porciones se trazan en sentido antihorario, por defecto comenzando desde el eje-x. [11]

**Ejemplo**

colors = ['cyan','lime','pink']

explode = [0,0,0]

labels = dsFelicidadAnual.index

sizes = dsFelicidadAnual.values

plt.figure(figsize=(7,7))

plt.title('% Anual PuntajeFelicidad', color='purple', fontsize=15)

# gráfico de torta

plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%')

plt.show()

|  |
| --- |
|  |
| Muestra de los datos para el gráfico. |

|  |
| --- |
|  |
| Ejemplo de matplotlib.pyplot.pie() |

## → Gráficos tipo Histograma

matplotlib.pyplot.hist()  
  
  
En estadística, un **histograma** es una representación gráfica de una variable en forma de barras, donde la superficie de cada barra es proporcional a la frecuencia de los valores representados.   
  
Sirven para obtener una "primera vista" general, o panorama, de la distribución de la población, o de la muestra, respecto a una característica, cuantitativa y continua (como una longitud o un peso). [12]

## → **Diagramas** de Dispersión **(Scatter)**

matplotlib.pyplot.scatter()  
  
  
Un **diagrama de dispersión** o gráfica de dispersión o gráfico de burbujas es un tipo de diagrama matemático que utiliza las coordenadas cartesianas para mostrar los valores de dos variables para un conjunto de datos.  
  
Se emplea cuando una o varias variables está bajo el control del experimentador. Si existe un parámetro que se incrementa o disminuye de forma sistemática por el experimentador, se le denomina *parámetro de control* o variable independiente y habitualmente se representa a lo largo del eje horizontal (eje de las abscisas). La variable medida o dependiente usualmente se representa a lo largo del eje vertical (eje de las ordenadas). Si no existe una variable dependiente, cualquier variable se puede representar en cada eje y el diagrama de dispersión mostrará el grado de correlación (no causalidad) entre las dos variables. [13]

## → **Diagramas** de Caja (Boxplot**)**

matplotlib.pyplot.boxplot()  
  
  
También conocido como diagrama de caja y bigote, box-plot o boxplot.

Es un método estandarizado para representar gráficamente una serie de datos numéricos a través de sus cuartiles. De esta manera, el diagrama de caja muestra a simple vista la mediana y los cuartiles de los datos, pudiendo también representar los [**valores atípicos**](https://es.wikipedia.org/wiki/Valor_atípico) de estos. [14]

|  |
| --- |
|  |
| Muestra de los datos para el gráfico. |

|  |
| --- |
|  |
| Ejemplo de matplotlib.pyplot.boxplot() |

# Anexo III: Librería SEABORN

Seaborn es una biblioteca open-source de visualización de datos de Python basada en **matplotlib**.  
Proporciona una interfaz de alto nivel para dibujar gráficos estadísticos atractivos e informativos.

Sitio oficial: [https://seaborn.pydata.org](https://seaborn.pydata.org/)  
GitHub oficial: https://github.com/mwaskom/seaborn

Se puede instalar mediante el comando:

pip install seaborn

Esta posee a su vez las siguientes dependencias:

python 3.6+  
numpy 1.13.3+  
scipy 1.0.1+  
pandas 0.22.0+  
matplotlib 2.1.2+

Se importa de la siguiente manera:

import seaborn as sns # visualización de datos

Se desarrolla en las próximas secciones un análisis de algunos de sus tipos de gráficos incluidos.

## → Gráficos de Barras

seaborn.barplot()  
  
  
Un gráfico de barras representa una estimación de la tendencia central de una variable numérica con la altura de cada rectángulo y proporciona alguna indicación de la incertidumbre en torno a esa estimación utilizando barras de error. Las gráficas de barras incluyen 0 en el rango de eje cuantitativo, y son una buena opción cuando 0 es un valor significativo para la variable cuantitativa, y desea hacer comparaciones con ella. [15]  
  
Los datos de entrada se pueden pasar en una variedad de formatos, que incluyen:  
- Vectores de datos representados como listas, matrices numpy u objetos de la serie pandas pasados directamente a los parámetros ‘x’, ‘y’, y/o ‘hue’.  
- Un DataFrame a lo largo (“long-form”), en cuyo caso las variables ‘x’, ‘y’ y ‘hue’ determinarán cómo se trazan los datos.  
- Un DataFrame a lo ancho ("wide-form"), de modo que se trazará cada columna numérica.  
- Una matriz o lista de vectores.  
  
En la mayoría de los casos, es posible usar objetos numpy o Python, pero los objetos pandas son preferibles porque los nombres asociados se usarán para anotar los ejes.

|  |
| --- |
|  |
| Gráfico de barras en forma horizontal con segmentación. |

|  |
| --- |
|  |
| Gráfico de barras en forma vertical. |

## → Gráficos de Pares

seaborn.pairplot()  
  
  
Grafica relaciones por pares en un conjunto de datos.  
  
De forma predeterminada, esta función creará una cuadrícula de Ejes de modo que cada variable numérica en los datos se compartirá en el eje y a través de una sola fila y en el eje x a través de una sola columna. Los ejes diagonales se tratan de manera diferente, dibujando un gráfico para mostrar la distribución univariada de los datos para la variable en esa columna. [16]  
  
También es posible mostrar un subconjunto de variables o trazar diferentes variables en las filas y columnas haciendo uso del parámetros vars, con lo nombres de variables del subconjunto.

|  |
| --- |
|  |
| Gráfico de pares. |

## → Gráficos tipo Joint

seaborn.jointplot()  
  
  
Dibuja una gráfica de dos variables con gráficas bivariadas y univariadas.  
  
Esta función proporciona una interfaz conveniente para la clase JointGrid, con varios tipos de gráficos enlatados. Se pretende que sea un envoltorio bastante ligero; Si necesita más flexibilidad, debe usar JointGrid directamente. [17]

|  |
| --- |
|  |
| Gráfico tipo Joint, subtipo KDE. |

|  |
| --- |
|  |
| Gráfico tipo Joint, subtipo Reg. |

## → Gráficos tipo **KDE**

seaborn.kdeplot()  
  
  
Ajusta y grafica una estimación de densidad, sea del tipo univariada o bivariada. [18]

Los datos de entrada se deben pasar como:  
- param ‘data’: un array unidimensional, mandatorio.  
- param ‘data2’: un array unidimensional, opcional, si se desea que la estimación sea bivariada.

|  |
| --- |
|  |
| Gráfico tipo KDE. |

## → Gráficos tipo **LM**

seaborn.lmplot()  
  
  
Esta función combina *regplot()* y *FacetGrid*. Está pensado como una interfaz conveniente para ajustar modelos de regresión a través de subconjuntos condicionales de un conjunto de datos.  
  
Cuando se piensa en cómo asignar variables a diferentes facetas, una regla general es que tiene sentido usar el parámetro *hue* para la comparación más importante, seguido de *col* y *row*.  
  
Los parámetros de esta función abarcan la mayoría de las opciones en *FacetGrid*, aunque puede haber casos ocasionales en los que desee utilizar esa clase y *regplot()* directamente. [19]

|  |
| --- |
|  |
| Gráfico tipo LM (Lineal Model). |

## → Gráficos tipo **Violín**

seaborn.violinplot()  
  
  
Un gráfico tipo violín tiene una función similar a los diagramas de cajas (boxplot).  
  
Muestra la distribución de datos cuantitativos a través de varios niveles de una (o más) variables categóricas para que esas distribuciones puedan compararse. A diferencia de un diagrama de caja, en el que todos los componentes del diagrama corresponden a puntos de datos reales, el diagrama de violín presenta una estimación de la densidad del núcleo de la distribución subyacente.  
  
Esta puede ser una forma efectiva y atractiva de mostrar múltiples distribuciones de datos a la vez, pero tenga en cuenta que el procedimiento de estimación está influenciado por el tamaño de la muestra, y los violines para muestras relativamente pequeñas pueden parecer engañosamente suaves.  
  
Los datos de entrada se pueden pasar en una variedad de formatos, que incluyen:  
- Vectores de datos representados como listas, matrices numpy u objetos de la serie pandas pasados directamente a los parámetros ‘x’, ‘y’, y/o ‘hue’.  
- Un DataFrame a lo largo ("long-form"), en cuyo caso las variables ‘x’, ‘y’ y ‘hue’ determinarán cómo se trazan los datos.  
- Un DataFrame a lo ancho (“wide-form”), de modo que se trazará cada columna numérica.  
- Una matriz o lista de vectores.  
  
En la mayoría de los casos, es posible usar objetos numpy o Python, pero los objetos pandas son preferibles porque los nombres asociados se usarán para anotar los ejes. Además, puede usar tipos categóricos para las variables de agrupación para controlar el orden de los elementos de la trama.  
  
Esta función siempre trata una de las variables como categórica y dibuja datos en posiciones ordinales (0, 1, ... n) en el eje relevante, incluso cuando los datos tienen un tipo numérico o de fecha. [20]

|  |
| --- |
|  |
| Gráfico tipo violín. |

# Referencias

1. UN: World Happiness Report

<https://worldhappiness.report/>

2. Datos Macro: Indice Mundial de la Felicidad

https://datosmacro.expansion.com/demografia/indice-felicidad

3. UN: Office on Drugs and Crime  
 <https://www.unodc.org/>

4. UN: Office on Drugs and Crime - Statistics  
 <https://www.unodc.org/unodc/en/data-and-analysis/statistics.html>  
  
5. Librería StatsModels  
 [https://www.statsmodels.org](https://www.statsmodels.org/)  
  
6. DatosMacro.com - Tasa de alfabetización  
 <https://datosmacro.expansion.com/demografia/tasa-alfabetizacion>  
  
7. Wikipedia: Anexo - Países por tasa de alfabetización  
 <https://es.wikipedia.org/wiki/Anexo:Países_por_tasa_de_alfabetización>  
  
8. CIA: The World Factbook - Literacy  
 <https://www.cia.gov/library/publications/the-world-factbook/fields/370.html>

9. Matplotlib: Manual oficial de Matplotlib, versión 3.2.1, pág. 13.  
 [https://matplotlib.org/3.2.1/Matplotlib.pd](https://matplotlib.org/3.2.1/Matplotlib.pdf)f  
  
10. Wikipedia: Gráfico Circular, sección ‘Características’.  
 <https://es.wikipedia.org/wiki/Gráfico_circular>  
  
11. Matplotlib: Manual oficial de Matplotlib, versión 3.2.1, pág. 944.  
 <https://matplotlib.org/3.2.1/Matplotlib.pdf>  
  
12. Wikipedia: Histograma.  
 <https://es.wikipedia.org/wiki/Histograma>  
  
13 Wikipedia: Diagrama de Dispersión.  
 <https://es.wikipedia.org/wiki/Diagrama_de_dispersión>  
  
14. Wikipedia: Diagrama de Caja.  
 <https://es.wikipedia.org/wiki/Diagrama_de_caja>  
  
15. Seaborn: Documentación Oficial - Barplot  
 <https://seaborn.pydata.org/generated/seaborn.barplot.html>  
  
16. Seaborn: Documentación Oficial - Pairplot  
 <https://seaborn.pydata.org/generated/seaborn.pairplot.html>  
  
17. Seaborn: Documentación Oficial - Jointplot  
 <https://seaborn.pydata.org/generated/seaborn.jointplot.html>  
  
18. Seaborn: Documentación Oficial - Kdeplot  
 <https://seaborn.pydata.org/generated/seaborn.kdeplot.html>  
  
19. Seaborn: Documentación Oficial – Lmplot  
 <https://seaborn.pydata.org/generated/seaborn.lmplot.html>  
  
20. Seaborn: Documentación Oficial - Violinplot  
 <https://seaborn.pydata.org/generated/seaborn.violinplot.html>