Jaime Francisco Barbosa Hernández ID:615152

SC3314 – Inteligencia Artificial Universidad de Monterrey Dr. Antonio Martínez Torteya

A1.2 Regresión lineal simple

La felicidad no es algo fácilmente mensurable, aun así, la publicación World Happiness Report (WHR) ha tratado desde el 2012 de medir los niveles de felicidad de la gran mayoría de los países del mundo. Los resultados que obtienen año con año se publican de forma abierta, así como los datos que lograron capturar.

La variable de mayor interés para este ejercicio es el nivel general de felicidad por país, evaluado en una escala del 0 al 10. Una de las variables que en dicho reporte han expuesto como relevante para determinar la felicidad de una nación es su producto interno bruto, o gross domestic product (GDP). Los datos de GDP por país se pueden descargar desde el sitio de The World Bank, una institución financiera internacional.

En este caso, te presento un archivo de nombre "A1.2 Felicidad y GDP.csv" (que podrás encontrar en la misma página de la plataforma donde descargaste esta plantilla), donde podrás encontrar el nivel de felicidad del último reporte del WHR (2022) y el GDP (2020) de los países para los que se cuenta con información. Los datos de niveles de felicidad se descargaron directamente del sitio de internet del WHR, y los datos de GDP se descargaron del sitio de internet de The World Bank.

La base de datos cuenta con la siguiente información:

- "Pais". Se describe el nombre del país.
- "Felicidad". Un número entre 0 y 10 que describe el nivel de felicidad.
- "GDP". Un número que describe el producto interno bruto.

Es momento de poner en práctica los conocimientos que hemos adquirido para encontrar un modelo de regresión lineal simple en el que relaciones la felicidad y el GDP de un país. Antes de continuar, pregúntate: ¿qué dirección crees que tendrá la asociación (a mayor GDP, mayor o menor felicidad)?

Desarrolla los siguientes puntos en una Jupyter Notebook, tratando, dentro de lo posible, que cada punto se trabaje en una celda distinta. Los comentarios en el código siempre son bienvenidos, de preferencia, aprovecha el markdown para generar cuadros de descripción que ayuden al lector a comprender el trabajo realizado.

1. Importa los datos del archivo "Felicidad y GDP.csv" a tu ambiente de trabajo. Por curiosidad, revisemos cuáles son los países más felices, así como los que tienen mayor GDP. Imprime en consola un resumen de 10 filas de la base de datos, previamente

ordenada de mayor a menor felicidad. Te recomiendo que hagas uso de la función "sort_values()", especificando como primer parámetro el nombre de la columna de interés, y el parámetro "ascending" con valor "False". Repite el proceso, pero ahora ordenando la base de datos de mayor a menor GDP.

In [31]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
Fdata=pd.read_csv("A1.2 Felicidad y GDP.csv")
print(Fdata.head(10))
print()
print()
print(Fdata.sort_values(by="GDP",ascending=False).head(10))
print()
print()
print(Fdata.sort_values(by="GDP",ascending=True).head(10))
```

```
         Pais  Felicidad           GDP
0     Finland     7.8210  2.718370e+11
1     Denmark     7.6362  3.560850e+11
2     Iceland     7.5575  2.171808e+10
3  Switzerland     7.5116  7.522480e+11
4  Netherlands     7.4149  9.138650e+11
5   Luxembourg     7.4040  7.335313e+10
6       Sweden     7.3843  5.414870e+11
7       Norway     7.3651  3.621980e+11
8       Israel     7.3638  4.071010e+11
9  New Zealand     7.1998  2.117350e+11


              Pais  Felicidad           GDP
15   United States     6.9768  2.089370e+13
70           China     5.5853  1.468770e+13
52           Japan     6.0389  5.040110e+12
13         Germany     7.0341  3.846410e+12
16  United Kingdom     6.9425  2.756900e+12
130         India     3.7771  2.667690e+12
19         France     6.6867  2.630320e+12
29          Italy     6.4667  1.892570e+12
14         Canada     7.0251  1.645420e+12
57     South Korea     5.9351  1.637900e+12


            Pais  Felicidad           GDP
111      Comoros     4.6086  1.223876e+09
89        Gambia     5.1636  1.830413e+09
135      Lesotho     3.5118  2.250718e+09
93       Liberia     5.1215  3.039983e+09
119     Eswatini     4.3961  3.984841e+09
134  Sierra Leone     3.5740  4.063289e+09
73     Montenegro     5.5468  4.780722e+09
129         Togo     4.1123  7.574637e+09
30        Kosovo     6.4551  7.716925e+09
62    Kyrgyzstan     5.8285  7.780875e+09
```

En este inciso se importan las librerías de pandas, numpy y matplotlib.pyplot que se usarán a lo largo del caso. Utilizando la funcion "read_csv" de pandas se lee la base de datos, para luego acomodarlas por GDP ascendente y descendente mediante la funcion "sort_values", y se imprimen los primeros 10 datos.

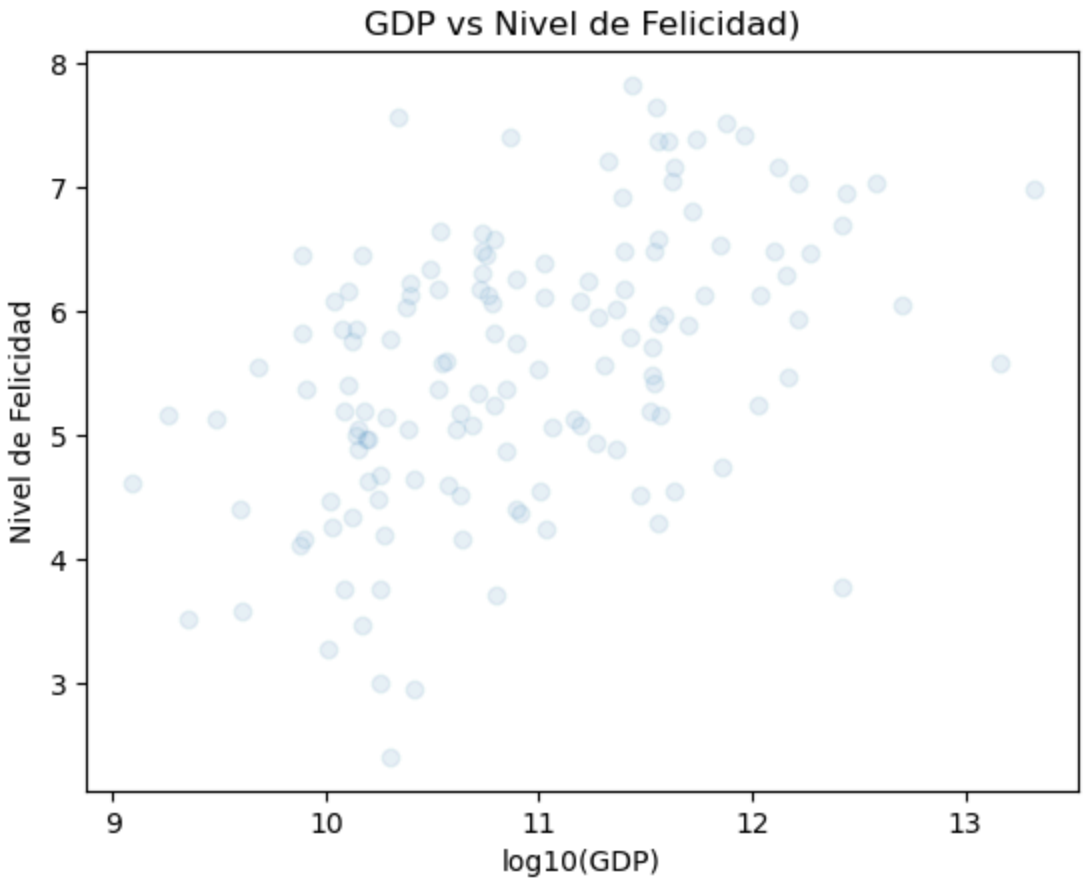
2. Grafica las observaciones, definiendo el valor de “y” como el nivel de felicidad y el valor de “x” como el logaritmo base 10 del GDP. Típicamente, cuando trabajamos con números tan grandes como lo es el GDP, es mucho más común encontrar asociaciones con el logaritmo base 10 de la variable, que con la variable misma. Esto no es un problema, siempre y cuando nuestra conclusión incluya este recordatorio. Es decir, si descubrimos que sí existe una asociación, diríamos que: “encontramos una asociación estadísticamente significativa entre los niveles de felicidad y el logaritmo base 10 del GDP”. La librería numpy tiene la función “log10()”, puedes usarla para transformar el GDP.

In [3]:
```
%matplotlib inline
x=np.log10(Fdata.GDP)
y=Fdata.Felicidad


plt.title("GDP vs Nivel de Felicidad)")
plt.xlabel("log10(GDP)")
plt.ylabel("Nivel de Felicidad")

plt.scatter(x, y,alpha=0.09)

plt.show()
```



En este inciso se define la variable "x" como el logaritmo base 10 del GDP, y la variable "y" como el nivel de felicidad, y se procede a hacer una gráfica de dispersión con dichas variables.

3. Calcula los valores óptimos de los dos coeficientes del modelo de regresión lineal simple. Realiza este proceso “a mano”, sin apoyarte con funciones preestablecidas de librerías de análisis de datos. Al finalizar, imprime en consola ambos valores. Siempre es bueno especificar qué es el valor que estamos imprimiendo, por lo que te recomiendo usar una sintaxis similar a: print(“B1 =”, B1).

In [4]:
```
Beta1_hat=sum((x-np.mean(x))*(y-np.mean(y)))/sum((x-np.mean(x))**2)
Beta0_hat=np.mean(y)-Beta1_hat*np.mean(x)
print("B1= ",Beta1_hat)
print("B0= ",Beta0_hat)
```

```
B1=  0.6281284658810408
B0=  -1.3023500570747277
```

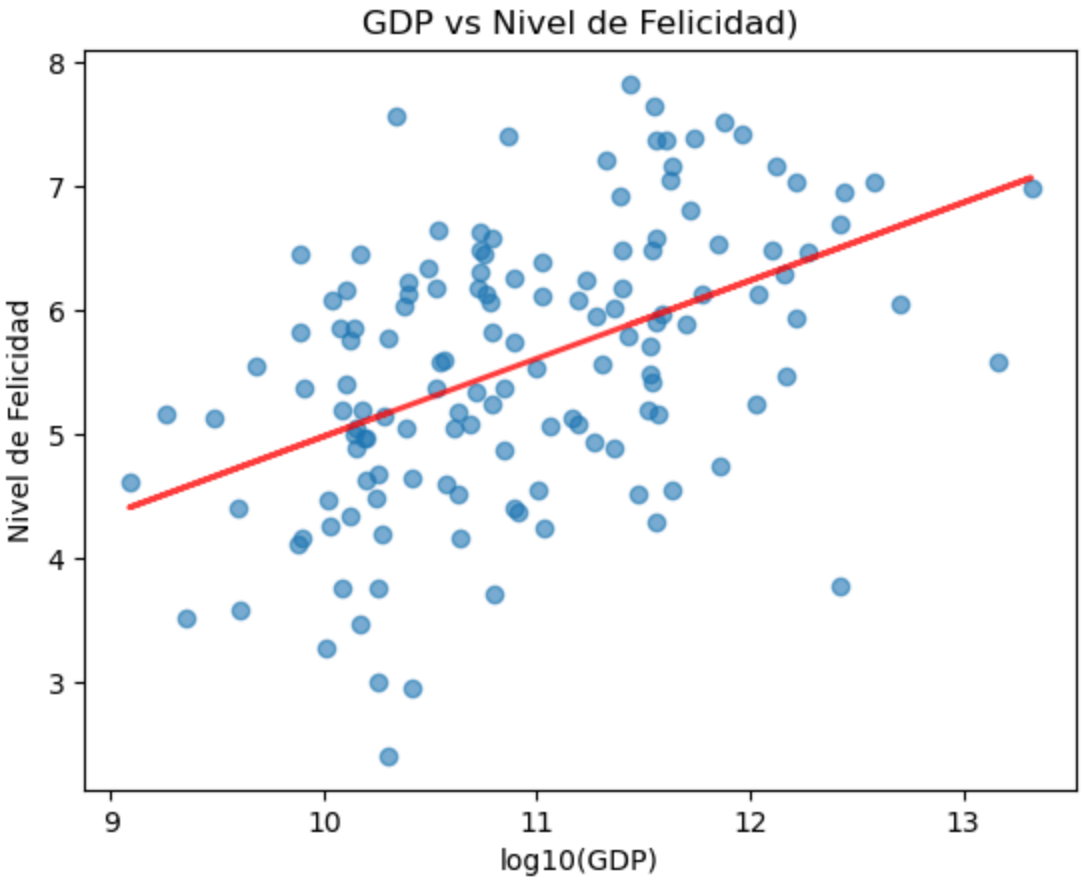
En este inciso, utilizando las funcion "mean" de numpy, se calcularon tanto Beta1 como Beta0. Estas ecuaciónes surgen al derivar parcialmente igualar a cero (puesto que se quiere reducir el error cuadrático), y resolver el sistema de ecuaciónes.

4. Realiza una nueva gráfica en la que se muestren tanto las observaciones, como la línea que representa el modelo de regresión lineal simple. Te recomiendo usar un valor de transparencia de 0.75, tanto para las observaciones como para la línea.

In [5]:
```
y_pred=Beta1_hat*x+Beta0_hat

plt.title("GDP vs Nivel de Felicidad)")
plt.xlabel("log10(GDP)")
plt.ylabel("Nivel de Felicidad")

plt.scatter(x, y,alpha=0.6)
plt.plot(x,y_pred,c="r",linewidth=2,alpha=0.75)
plt.show()
%matplotlib inline
```



En este siguiente inciso, se crea la variable y_pred, la cual representa el modelo de regresión lineal simple, usando los valores de Beta1 y Beta0 que se calcularon previamente. Luego, se grafican tanto los datos/observaciones con una gráfica de dispercion, y la recta de nuestro modelo.

5. Calcula el RSS del modelo e imprímelo en la consola

In [6]:
```
E=(y-y_pred)**2
RSS=sum(E)
```

```
print("RSS= ",RSS)
```

```
RSS=  131.3738317732635
```

En este inciso se calcula el RSS, que nos será útil para obtener el error estandar, entre otras cosas.

6. Calcula el error estándar e intervalo de confianza de $\beta 1$, e imprime dichos valores en la consola. Siempre es bueno especificar qué es el valor que estamos imprimiendo, por lo que te recomiendo usar una sintaxis similar a: print("SE =", SE). Asimismo, determina si la asociación entre la felicidad y el GDP es significativa en esta población, imprimiendo en consola un mensaje que claramente explique el por qué de la afirmación, evidenciado por alguna métrica calculada.

In [15]:
```
n=len(x)
den1=sum((x-np.mean(x))**2)
SE_B1=np.sqrt(RSS/((n-2)*den1))
print("SE(B1)= ",SE_B1)
print()

import scipy.stats as st
per = st.t.interval(confidence = 0.95, df = n-2)[1]
CIlow = Beta1_hat - per*SE_B1
CIhigh = Beta1_hat+ per*SE_B1
print("Inervalo de confianza: (",CIlow,",",CIhigh,")")
print()
print("Debido a que el error estandandar es bajo comparado con lo calculado con Bet
print()
print("Además, mediente el intervalo de confianza, se puede decir que es 95% probab
print(CIlow,"-",CIhigh,".")
print()
```

```
SE(B1)=  0.09983378435340727

Inervalo de confianza: ( 0.4307393313073311 , 0.8255176004547504 )

Debido a que el error estandandar es bajo comparado con lo calculado con Beta 1, se
puede decir que no hay tanta variabilidad en los datos.

Además, mediente el intervalo de confianza, se puede decir que es 95% probable que l
a tasa de cambio entre el logaritmo base 10 del GDP y el nivel de felicidad sea posi
tiva, es decir, a mayor GDP, mayor nivel de feilicidad, y esta tasa o razón de cambi
o probablemente está entre
0.4307393313073311 - 0.8255176004547504 .
```

7. Calcula el residual standard error y la R^2 del modelo, e imprime dichos valores en la consola. Para el cálculo de R^2, te recomiendo primero calcular el total sum of squares, o TSS. Para el cálculo del mismo, pon mucha atención al orden de los paréntesis, pues no es lo mismo sumar el cuadrado de múltiples valores, que sumar múltiples valores y elevar el resultado al cuadrado. Adicionalmente, agrega un comentario, imprimiéndolo en consola, sobre tu opinión del valor de R^2 obtenido con el modelo.

In [27]:
```
RSE=np.sqrt((1/(n-2))*sum((y-y_pred)**2))
print("RSE= ",RSE)
print()
TSS=sum((y-np.mean(y))**2)
R2=1-(RSS/TSS)
print("R2=", R2)
print()
print("El R^2 de 0.2216 nos da a entender que los datos no siguen precisamente la f
```

```
RSE=  0.9721807858537376

R2= 0.22166361654970657

El R^2 de 0.2216 nos da a entender que los datos no siguen precisamente la forma de
un modelo lineal, solo una pequeña porción lo hace
```

8. Finalmente, usa la función “OLS()” de la librería statsmodels.api para verificar que los resultados de todos los puntos anteriores son los esperados. Si el resumen de los resultados te entrega un p-value = 0.000, puedes observar el resultado con muchas más cifras significativas usando la función “pvalues” de la misma librería. Por ejemplo: si los resultados de ajustar el modelo los almacenaste en una variable de nombre “var”, puedes usar la siguiente sintaxis: var.pvalues. Por default, el RSE no se muestra en el resumen, para revisarlo utiliza la línea de código print(var.scale**.5), de nuevo asumiendo que almacenaste el resultado en una variable de nombre “var”.

In [30]:
```
import statsmodels.api as sm
model = sm.OLS(y,sm.add_constant(x))
results = model.fit()
print(results.summary())
print("RSE= ",results.scale**.5)
```

OLS Regression Results

Dep. Variable:	Felicidad	R-squared:	0.222
Model:	OLS	Adj. R-squared:	0.216
Method:	Least Squares	F-statistic:	39.59
Date:	Sun, 17 Aug 2025	Prob (F-statistic):	3.83e-09
Time:	18:10:46	Log-Likelihood:	-195.09
No. Observations:	141	AIC:	394.2
Df Residuals:	139	BIC:	400.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1.3024	1.094	-1.191	0.236	-3.465	0.860
GDP	0.6281	0.100	6.292	0.000	0.431	0.826

Omnibus:	2.648	Durbin-Watson:	0.455
Prob(Omnibus):	0.266	Jarque-Bera (JB):	2.523
Skew:	-0.326	Prob(JB):	0.283
Kurtosis:	2.944	Cond. No.	148.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
RSE= 0.9721807858537376