



Instituto Politécnico Nacional



Centro de Investigación en
Computación

Introducción a CHATGPT con Python

Práctica 102

Prof. Alan Badillo Salas

Agosto 2023

Víctor Manuel Fierro Hernández

victor-fierro@hotmail.com

1. INTRODUCCIÓN

La práctica 102 tiene como propósito evaluar la capacidad de CHATGPT con respecto a análisis estadísticos y análisis de información. Combinando librerías OCR de Python para la extracción de texto de archivos PDF, se pueden generar aplicaciones útiles, eficientes, rápidas y de aplicabilidad real.

A través de algunas preguntas realizadas a la herramienta, se evaluarán los resultados obtenidos y se proporcionarán comentarios y conclusiones considerando expectativas previas.

2. JUSTIFICACIÓN

A fin de incorporar CHATGPT como una herramienta de apoyo en actividades cotidianas, principalmente en el área de desarrollo de software, es requerido evaluar su nivel de comprensión tanto de datos numéricos, como de texto. Por esta razón, se evaluará el desempeño de CHATGPT para el análisis estadísticos y análisis de texto.

3. DESARROLLO

P102.1 - Redacción de Correos Electrónicos

Definición 1. Solicita a CHATGPT que genere el código para abrir un archivo weight-height.csv desde la URL

<https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/weight-height.csv> usando Pandas

Resultado

Genera el código para abrir un archivo weight-height.csv desde la URL

<https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/weight-height.csv> usando Pandas

Para abrir un archivo CSV desde una URL utilizando Pandas, primero debes asegurarte de tener Pandas instalado en tu entorno de Python. Si aún no lo tienes instalado, puedes hacerlo ejecutando el siguiente comando en tu terminal o consola:

```
```\n\npip install pandas\n```
```

Una vez que tengas Pandas instalado, puedes usar el siguiente código para abrir el archivo "weight-height.csv" desde la URL proporcionada:

```
```python\nimport pandas as pd\n\nurl = "https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/weight-height.csv"\ndf = pd.read_csv(url)\n```
```

```
# Ahora puedes trabajar con el DataFrame "df"
# Por ejemplo, puedes imprimir las primeras filas del DataFrame para ver los datos:
print(df.head())
'''
```

Este código importa Pandas y utiliza la función `pd.read_csv()` para cargar el archivo CSV desde la URL. Luego, el DataFrame resultante se almacena en la variable `df`. Puedes usar `print(df.head())` para imprimir las primeras filas del DataFrame y ver cómo se ven los datos cargados.

Recuerda que necesitas estar conectado a Internet para poder acceder a la URL y descargar el archivo CSV.

```
import pandas as pd

url = "https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/weight-height.csv"
df = pd.read_csv(url)

# Ahora puedes trabajar con el DataFrame "df"
# Por ejemplo, puedes imprimir las primeras filas del DataFrame para ver los datos:
print(df.head())
#print(df)
```

	Gender	Height	Weight
0	Male	73.847017	241.893563
1	Male	68.781904	162.310473
2	Male	74.110105	212.740856
3	Male	71.730978	220.042470
4	Male	69.881796	206.349801

Fig 2. Resultado del código ejecutado en Google Colab

Definición 2. Solicita a CHATGPT que genere el código que imprima 10 valores aleatorios del dataset anterior usando la función sample de Pandas

Gender Height Weight 0 Male 73.847017 241.893563 1 Male 68.781904 162.310473 2 Male 74.110105 212.740856 3 Male 71.730978 220.042470 4 Male 69.881796 206.349801

Resultado

Genera el código que imprima 10 valores aleatorios del dataset siguiente, usando la función sample de Pandas: Gender Height Weight 0 Male 73.847017 241.893563 1 Male 68.781904 162.310473 2 Male 74.110105 212.740856 3 Male 71.730978 220.042470 4 Male 69.881796 206.349801 si usando la función sample de Pandas

Para imprimir 10 valores aleatorios del dataset utilizando la función `sample` de Pandas, debes primero crear un DataFrame con los datos proporcionados y luego aplicar la función `sample`. Aquí está el código:

```
```python
import pandas as pd

Datos proporcionados
data = {
 'Gender': ['Male', 'Male', 'Male', 'Male', 'Male'],
 'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796],
 'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801]
}

Crear el DataFrame
df = pd.DataFrame(data)

Utilizar la función sample para obtener 10 valores aleatorios
num_samples = 10
random_samples = df.sample(num_samples)

Imprimir los valores aleatorios
print(random_samples)
```
```

El código crea un DataFrame utilizando los datos proporcionados. Luego, utiliza la función `sample` de Pandas con el argumento `num_samples` establecido en 10 para obtener 10 filas aleatorias del DataFrame original. Finalmente, imprime los valores aleatorios utilizando `print(random_samples)`. Cada vez que ejecutes el código, obtendrás una muestra diferente de 10 filas aleatorias del DataFrame.

```
[ ] # Crear el DataFrame
    #df = pd.DataFrame(data)

    # Utilizar la función sample para obtener 10 valores aleatorios
    num_samples = 10
    random_samples = df.sample(num_samples)

    # Imprimir los valores aleatorios
    print(random_samples)
```

| | Gender | Height | Weight |
|------|--------|-----------|------------|
| 31 | Male | 74.033808 | 212.910225 |
| 6671 | Female | 64.081270 | 157.589141 |
| 1990 | Male | 67.418241 | 166.542143 |
| 4713 | Male | 69.145775 | 184.487834 |
| 839 | Male | 70.046953 | 190.002405 |
| 1423 | Male | 67.192152 | 193.788816 |
| 1724 | Male | 67.140186 | 188.848499 |
| 876 | Male | 70.242866 | 192.796113 |
| 9373 | Female | 61.647397 | 111.927796 |
| 9749 | Female | 68.059839 | 176.244420 |

Fig 2. Resultado del código ejecutado en Google Colab

Definición 3. Solicita a CHATGPT que genere el código que describa la columna Gender que es categórica, pide que extraiga los valores únicos y sus conteos.

Resultado

Describe la columna Gender que es categórica, y extrae los valores únicos y sus conteos

La columna "Gender" es una columna categórica que representa el género de las personas en el dataset. Al ser categórica, esta columna contiene un conjunto finito de categorías o etiquetas que representan diferentes valores para el género, como "Male" (hombre) y posiblemente otros valores como "Female" (mujer) si el dataset incluye datos de ambos géneros.

Para extraer los valores únicos y sus conteos de la columna "Gender", podemos utilizar el método `value_counts()` de Pandas. Aquí está el código para hacerlo:

```
```python
import pandas as pd

Datos proporcionados
data = {
 'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male', 'Female'],
 'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030, 64.015163, 72.224406, 70.824881, 63.379272],
 'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912, 113.132207, 140.998265, 158.503610, 128.871867]
}

Crear el DataFrame
df = pd.DataFrame(data)

Extraer valores únicos y sus conteos de la columna 'Gender'
gender_counts = df['Gender'].value_counts()

Imprimir los valores únicos y sus conteos
print(gender_counts)
```

```
'''
```

Este código generará una salida que muestra los valores únicos presentes en la columna "Gender" y la cantidad de veces que aparece cada valor en el dataset. La salida será algo así:

```
'''
```

```
Male 6
Female 4
Name: Gender, dtype: int64
'''
```

Esto indica que hay 6 registros con el valor "Male" y 4 registros con el valor "Female" en la columna "Gender". Los valores únicos y sus conteos nos proporcionan una visión general de la distribución de género en el dataset.

```
#import pandas as pd

Datos proporcionados
#data = {
'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male', 'Female'],
'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030, 64.015163, 72.224406, 70.824881, 63.379272],
'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912, 113.132207, 140.998265, 158.503610, 128.871867]
#}

Crear el DataFrame
#df = pd.DataFrame(data)

Extraer valores únicos y sus conteos de la columna 'Gender'
gender_counts = df['Gender'].value_counts()

Imprimir los valores únicos y sus conteos
print(gender_counts)
```

```
Male 5000
Female 5000
Name: Gender, dtype: int64
```

**Definición 4. Solicita a CHATGPT que genere el código que describa la columna Height que es numérica y obtenga los estadísticos: total, mínimo, máximo, promedio, desviación estándar, intervalo de confianza al 95%, y los cuartiles.**

## **Resultado**

Genera el código que describa la columna Height que es numérica y obtenga los estadísticos: total, mínimo, máximo, promedio, desviación estándar, intervalo de confianza al 95%, y los cuartiles

Para describir la columna "Height", que es numérica, podemos utilizar el método `describe()` de Pandas. Esto nos proporcionará un resumen estadístico que incluye el recuento total, el valor mínimo, el valor máximo, el promedio (media), la desviación estándar, los cuartiles y otros estadísticos. También podemos calcular el intervalo de confianza al 95% utilizando la librería `scipy`.

Aquí está el código para hacerlo:

```
```python
import pandas as pd
import numpy as np
from scipy import stats

# Datos proporcionados
data = {
    'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male', 'Female'],
    'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030,
64.015163, 72.224406, 70.824881, 63.379272],
    'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912,
113.132207, 140.998265, 158.503610, 128.871867]
}

# Crear el DataFrame
df = pd.DataFrame(data)

# Descripción estadística de la columna 'Height'
height_stats = df['Height'].describe()
```

```
# Calcular el intervalo de confianza al 95%

confidence_interval = stats.t.interval(0.95, len(df['Height']) - 1, loc=np.mean(df['Height']),
scale=stats.sem(df['Height']))

# Imprimir los resultados

print("Descripción estadística de la columna 'Height':")

print(height_stats)

print("\nIntervalo de confianza al 95%: ({:.3f}, {:.3f})".format(confidence_interval[0],
confidence_interval[1]))

...
```

La salida mostrará la descripción estadística de la columna "Height", que incluye el recuento total, el valor mínimo, el valor máximo, el promedio (media), la desviación estándar y los cuartiles (25%, 50% y 75%). También mostrará el intervalo de confianza al 95%.

Es importante mencionar que los estadísticos y el intervalo de confianza pueden variar si se usan datos diferentes a los proporcionados en el ejemplo.

```
#import pandas as pd
import numpy as np
from scipy import stats

# Descripción estadística de la columna 'Height'
height_stats = df['Height'].describe()

# Calcular el intervalo de confianza al 95%
confidence_interval = stats.t.interval(0.95, len(df['Height']) - 1, loc=np.mean(df['Height']), scale=stats.sem(df['Height']))

# Imprimir los resultados
print("Descripción estadística de la columna 'Height':")
print(height_stats)
print("\nIntervalo de confianza al 95%: ({:.3f}, {:.3f})".format(confidence_interval[0], confidence_interval[1]))
```

Descripción estadística de la columna 'Height':
 count 10000.000000
 mean 66.367560
 std 3.847528
 min 54.263133
 25% 63.505620
 50% 66.318070
 75% 69.174262
 max 78.998742
 Name: Height, dtype: float64

Intervalo de confianza al 95%: (66.292, 66.443)

Definición 5. Solicita a CHATGPT que genere el código que describa la columna Weight que es numérica y obtenga los estadísticos: total, mínimo, máximo, promedio, desviación estándar, intervalo de confianza al 95%, y los cuartiles.

Resultado

Genere el código que describa la columna Weight que es numérica y obtenga los estadísticos: total, mínimo, máximo, promedio, desviación estándar, intervalo de confianza al 95%, y los cuartiles

Para describir la columna "Weight", que es numérica, y obtener los estadísticos solicitados, utilizaremos el método `describe()` de Pandas para obtener el recuento, mínimo, máximo, promedio y desviación estándar. Luego, calcularemos los cuartiles utilizando la función `quantile()` de Pandas. Finalmente, calcularemos el intervalo de confianza al 95% utilizando la librería `scipy`.

Aquí tienes el código:

```
```python
import pandas as pd
import numpy as np
from scipy import stats

Datos proporcionados
data = {
 'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male',
 'Female'],
 'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030,
 64.015163, 72.224406, 70.824881, 63.379272],
 'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912,
 113.132207, 140.998265, 158.503610, 128.871867]
}

Crear el DataFrame
df = pd.DataFrame(data)

Descripción estadística de la columna 'Weight'
weight_stats = df['Weight'].describe()

Calcular el intervalo de confianza al 95%
confidence_interval = stats.t.interval(0.95, len(df['Weight']) - 1, loc=np.mean(df['Weight']),
scale=stats.sem(df['Weight']))

Calcular los cuartiles
quartiles = df['Weight'].quantile([0.25, 0.5, 0.75])

Imprimir los resultados
print("Descripción estadística de la columna 'Weight':")
print(weight_stats)
print("\nIntervalo de confianza al 95%: ({:.3f}, {:.3f})".format(confidence_interval[0],
confidence_interval[1]))
print("\nCuartiles:")
print(quartiles)
```
```

La salida mostrará la descripción estadística de la columna "Weight", que incluye el recuento total, el valor mínimo, el valor máximo, el promedio (media), la desviación estándar y los cuartiles (25%, 50% y 75%). También mostrará el intervalo de confianza al 95% para la columna "Weight".

Recuerda que los resultados pueden variar si se usan datos diferentes a los proporcionados en el ejemplo.

```
# Descripción estadística de la columna 'Weight'
weight_stats = df['Weight'].describe()

# Calcular el intervalo de confianza al 95%
confidence_interval = stats.t.interval(0.95, len(df['Weight']) - 1, loc=np.mean(df['Weight']), scale=stats.sem(df['Weight']))

# Calcular los cuartiles
quartiles = df['Weight'].quantile([0.25, 0.5, 0.75])

# Imprimir los resultados
print("Descripción estadística de la columna 'Weight':")
print(weight_stats)
print("\nIntervalo de confianza al 95%: ({:.3f}, {:.3f})".format(confidence_interval[0], confidence_interval[1]))
print("\nCuartiles:")
print(quartiles)
```

```
Descripción estadística de la columna 'Weight':
count    10000.000000
mean      161.440357
std        32.108439
min        64.700127
25%       135.818051
50%       161.212928
75%       187.169525
max       269.989699
Name: Weight, dtype: float64

Intervalo de confianza al 95%: (160.811, 162.070)

Cuartiles:
0.25    135.818051
0.50    161.212928
0.75    187.169525
Name: Weight, dtype: float64
```

Definición 6. Solicita a CHATGPT que genere el código que grafique el histograma de la columna Weight usando Seaborn

Resultado

Para graficar el histograma de la columna "Weight" utilizando Seaborn, necesitas seguir los mismos pasos que en el ejemplo anterior, asegurándote de tener Seaborn y Matplotlib instalados en tu entorno de Python. Si ya tienes estas bibliotecas instaladas, puedes utilizar el siguiente código:

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

Datos proporcionados
data = {
 'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male', 'Female'],
 'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030, 64.015163, 72.224406, 70.824881, 63.379272],
 'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912, 113.132207, 140.998265, 158.503610, 128.871867]
}

Crear el DataFrame
df = pd.DataFrame(data)

Graficar el histograma de la columna 'Weight' usando Seaborn
sns.histplot(data=df, x='Weight', kde=True)
```

```
Agregar etiquetas y título
plt.xlabel('Weight')
plt.ylabel('Frequency')
plt.title('Histogram of Weight')
```

```
Mostrar el gráfico
plt.show()
```
```

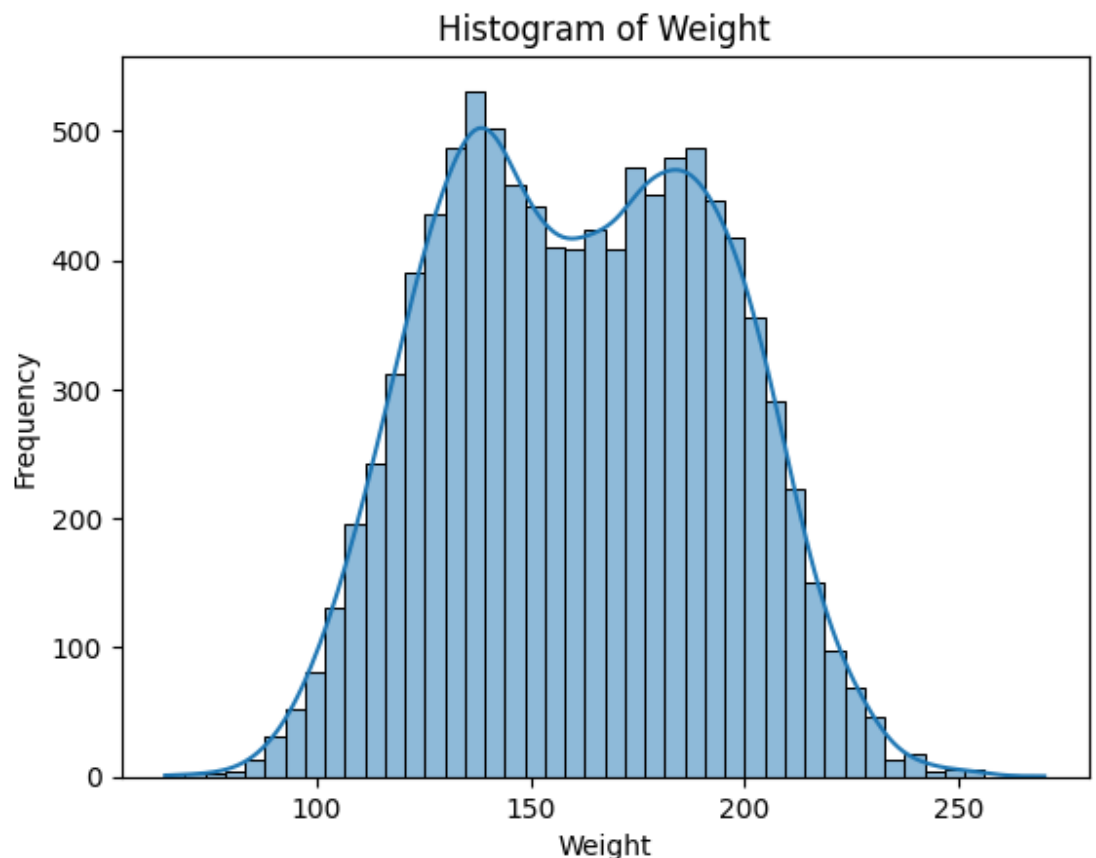
Este código utilizará Seaborn para generar el histograma de la columna "Weight". La opción `kde=True` agrega una estimación de la densidad de kernel para obtener una representación suave del histograma. Luego, agregamos etiquetas y un título al gráfico para hacerlo más informativo. Finalmente, utilizamos `plt.show()` para mostrar el gráfico.

El histograma mostrará la distribución de pesos en el dataset y te permitirá tener una idea de cómo están distribuidos los valores de la columna "Weight". Recuerda que los resultados pueden variar si usas datos diferentes a los proporcionados en el ejemplo.

```
# Graficar el histograma de la columna 'Weight' usando Seaborn
sns.histplot(data=df, x='Weight', kde=True)

# Agregar etiquetas y título
plt.xlabel('Weight')
plt.ylabel('Frequency')
plt.title('Histogram of Weight')

# Mostrar el gráfico
plt.show()
```



Definición 7. Solicita a CHATGPT que genere el código que grafique el histograma de la columna Height usando Seaborn

Resultado

Genera el código que grafique el histograma de la columna Height usando Seaborn

Para graficar el histograma de la columna "Height" utilizando Seaborn, primero debes asegurarte de tener Seaborn y Matplotlib instalados en tu entorno de Python. Si aún no los tienes instalados, puedes hacerlo ejecutando los siguientes comandos en tu terminal o consola:

```
```\n\npip install seaborn\npip install matplotlib\n```
```

Una vez que tengas las bibliotecas instaladas, puedes utilizar el siguiente código para graficar el histograma:

```
```python\nimport pandas as pd\nimport seaborn as sns\nimport matplotlib.pyplot as plt\n\n# Datos proporcionados\ndata = {\n    'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male',\n    'Female'],\n    'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030,\n    64.015163, 72.224406, 70.824881, 63.379272],\n    'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912,\n    113.132207, 140.998265, 158.503610, 128.871867]\n}\n\n# Crear el DataFrame\ndf = pd.DataFrame(data)\n\n# Graficar el histograma de la columna 'Height' usando Seaborn\nsns.histplot(data=df, x='Height', kde=True)\n\n# Agregar etiquetas y título\nplt.xlabel('Height')\nplt.ylabel('Frequency')\nplt.title('Histogram of Height')\n\n# Mostrar el gráfico\nplt.show()\n```
```

Este código utilizará Seaborn para generar el histograma de la columna "Height". La opción `kde=True` agrega una estimación de la densidad de kernel para obtener una representación suave del histograma. Luego, agregamos etiquetas y un título al gráfico para hacerlo más informativo. Finalmente, utilizamos `plt.show()` para mostrar el gráfico.

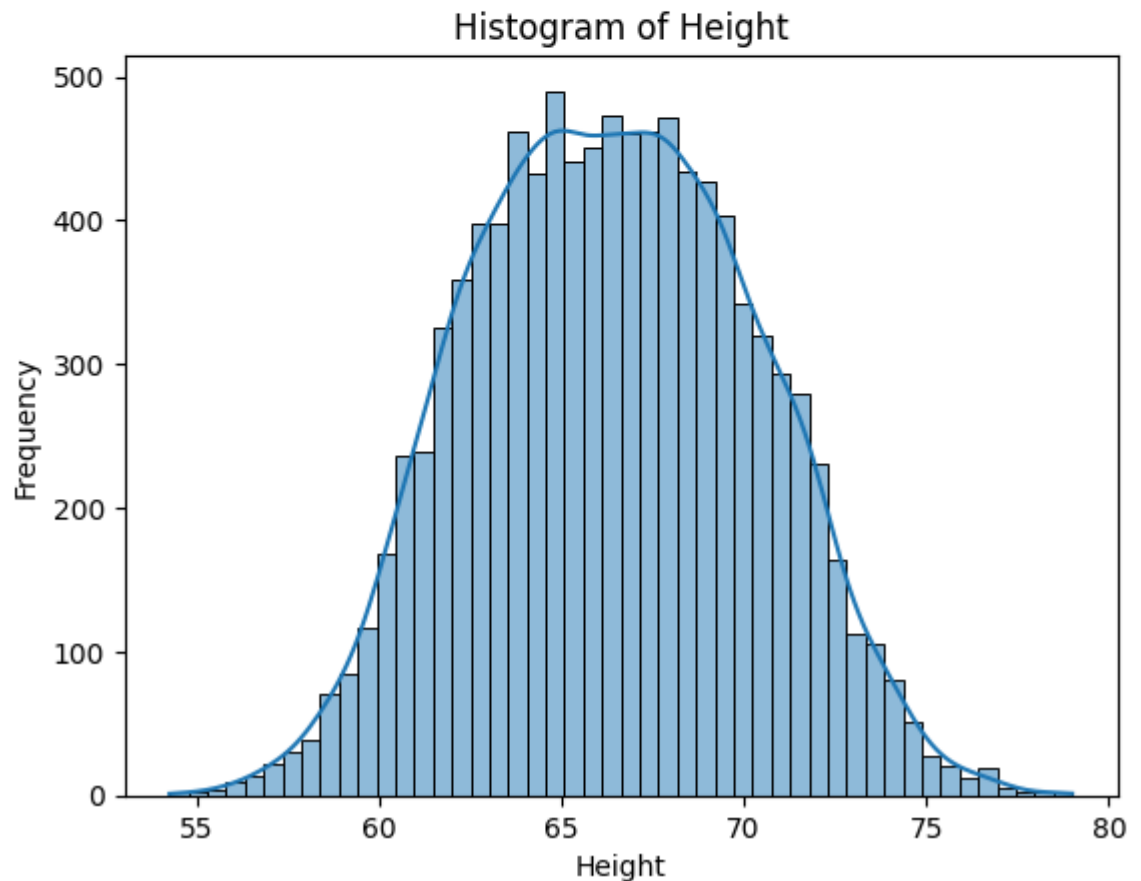
El histograma mostrará la distribución de alturas en el dataset y te permitirá tener una idea de cómo están distribuidos los valores de la columna "Height". Recuerda que los resultados pueden variar si usas datos diferentes a los proporcionados en el ejemplo.

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

# Graficar el histograma de la columna 'Height' usando Seaborn
sns.histplot(data=df, x='Height', kde=True)

# Agregar etiquetas y título
plt.xlabel('Height')
plt.ylabel('Frequency')
plt.title('Histogram of Height')

# Mostrar el gráfico
plt.show()
```



Definición 8. Solicita a CHATGPT que genere el código que grafique la correlación entre las columnas Height y Weight usando Seaborn

Resultado

Genera el código que grafique la correlación entre las columnas Height y Weight usando Seaborn

Para graficar la correlación entre las columnas "Height" y "Weight" utilizando Seaborn, puedes usar un gráfico de dispersión (scatter plot) con un ajuste lineal. Seaborn ofrece la función `sns.scatterplot()` para realizar esta tarea. Aquí tienes el código para hacerlo:

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

Datos proporcionados
data = {
 'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male',
 'Female'],
 'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030,
 64.015163, 72.224406, 70.824881, 63.379272],
 'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912,
 113.132207, 140.998265, 158.503610, 128.871867]
}

Crear el DataFrame
df = pd.DataFrame(data)

Graficar la correlación entre 'Height' y 'Weight' usando Seaborn
sns.scatterplot(data=df, x='Height', y='Weight')

Agregar etiquetas y título
plt.xlabel('Height')
plt.ylabel('Weight')
plt.title('Correlation between Height and Weight')

Mostrar el gráfico
plt.show()
```
```

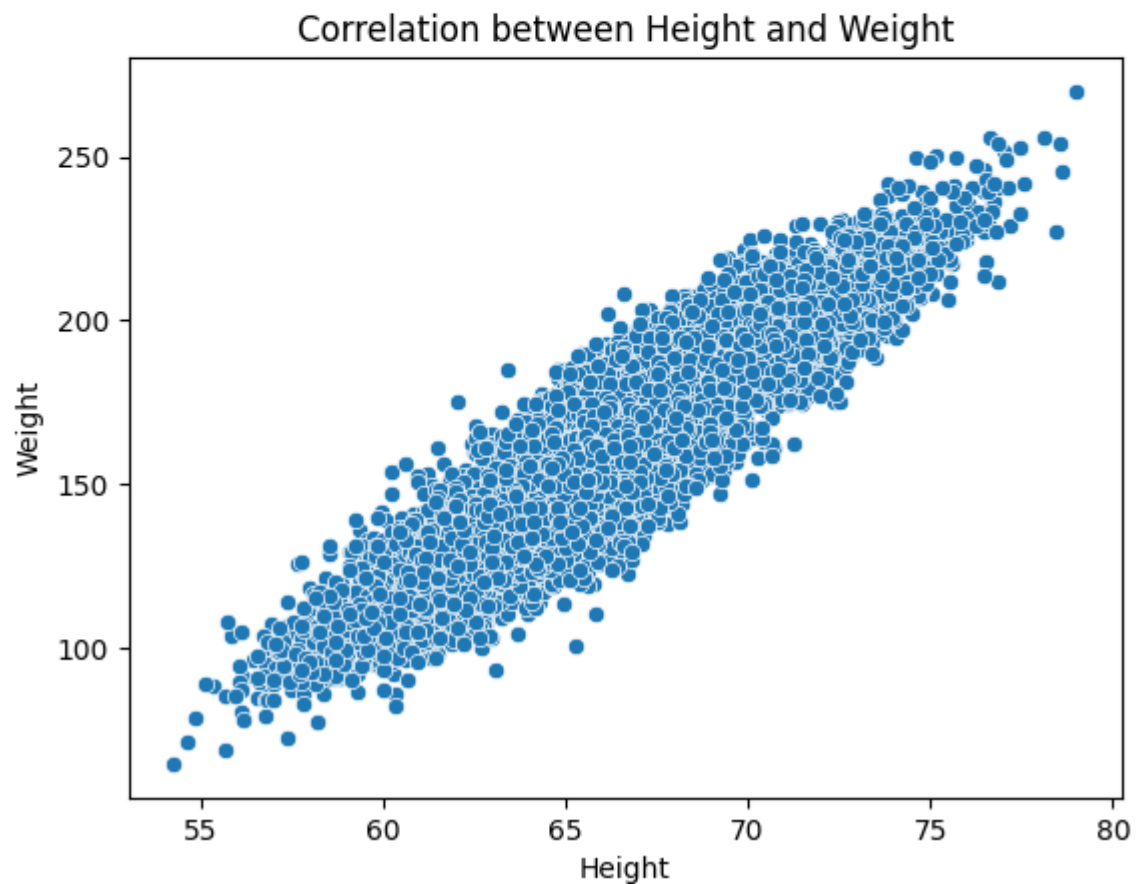
Este código utilizará Seaborn para generar un gráfico de dispersión que muestra la correlación entre las columnas "Height" y "Weight". Cada punto en el gráfico representa un par de valores (altura y peso) para una observación en el dataset. Si los puntos se agrupan en una dirección diagonal, indica una correlación positiva entre las dos variables; si se agrupan en una dirección diagonal en sentido opuesto, indica una correlación negativa. Si los puntos están dispersos sin una tendencia clara, indica una correlación débil o nula.

El gráfico de dispersión te permite visualizar la relación entre las dos variables y determinar si existe una correlación entre ellas. Recuerda que los resultados pueden variar si usas datos diferentes a los proporcionados en el ejemplo.

```
# Graficar la correlación entre 'Height' y 'Weight' usando Seaborn
sns.scatterplot(data=df, x='Height', y='Weight')

# Agregar etiquetas y título
plt.xlabel('Height')
plt.ylabel('Weight')
plt.title('Correlation between Height and Weight')

# Mostrar el gráfico
plt.show()
```



Definición 9. Solicita a CHATGPT que genere el código que obtenga la correlación entre las columnas Height y Weight usando corr de Numpy

Resultado

Genere el código que obtenga la correlación entre las columnas Height y Weight usando corr de Numpy

Para obtener la correlación entre las columnas "Height" y "Weight" utilizando la función `corr` de NumPy, necesitas tener los datos en forma de matrices NumPy. A continuación, te mostraré cómo convertir las columnas en matrices y calcular la correlación:

```
```python
import pandas as pd
```

```

import numpy as np

Datos proporcionados
data = {
 'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male',
 'Female'],
 'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030,
 64.015163, 72.224406, 70.824881, 63.379272],
 'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912,
 113.132207, 140.998265, 158.503610, 128.871867]
}

Crear el DataFrame
df = pd.DataFrame(data)

Obtener las columnas 'Height' y 'Weight' como matrices NumPy
heights = df['Height'].to_numpy()
weights = df['Weight'].to_numpy()

Calcular la correlación utilizando la función corrcoef de NumPy
correlation_matrix = np.corrcoef(heights, weights)

La correlación entre 'Height' y 'Weight' está en la posición (0, 1) de la matriz
correlation = correlation_matrix[0, 1]

Imprimir el resultado
print("Correlation between Height and Weight:", correlation)
'''

```

Este código calculará la correlación entre las columnas "Height" y "Weight" usando la función `corrcoef` de NumPy, que toma dos matrices NumPy y devuelve una matriz de correlación. La correlación entre las dos columnas estará en la posición (0, 1) de la matriz resultante.

Recuerda que los resultados pueden variar si usas datos diferentes a los proporcionados en el ejemplo. Además, si ya tienes los datos como matrices NumPy, puedes saltarte el paso de convertir las columnas en matrices.

```

Obtener las columnas 'Height' y 'Weight' como matrices NumPy
heights = df['Height'].to_numpy()
weights = df['Weight'].to_numpy()

Calcular la correlación utilizando la función corrcoef de NumPy
correlation_matrix = np.corrcoef(heights, weights)

La correlación entre 'Height' y 'Weight' está en la posición (0, 1) de la matriz
correlation = correlation_matrix[0, 1]

Imprimir el resultado
print("Correlation between Height and Weight:", correlation)

```

Correlation between Height and Weight: 0.9247562987409146



**Definición 10. Solicita a CHATGPT que genere el código que reporte el valor promedio de la columna Height donde Gender es Male y donde Gender es Female y repita lo mismo para la columna Weight**

## Resultado

Genere el código que reporte el valor promedio de la columna Height donde Gender es Male y donde Gender es Female y repita lo mismo para la columna Weight

Para calcular el valor promedio de las columnas "Height" y "Weight" separando por el género (Male y Female), puedes utilizar el método `groupby()` de Pandas junto con la función `mean()` para obtener los promedios. Aquí tienes el código para hacerlo:

```
```python
import pandas as pd

# Datos proporcionados
data = {
    'Gender': ['Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male', 'Female'],
    'Height': [73.847017, 68.781904, 74.110105, 71.730978, 69.881796, 63.468030, 64.015163, 72.224406, 70.824881, 63.379272],
    'Weight': [241.893563, 162.310473, 212.740856, 220.042470, 206.349801, 132.818912, 113.132207, 140.998265, 158.503610, 128.871867]
}

# Crear el DataFrame
df = pd.DataFrame(data)

# Calcular el valor promedio de 'Height' y 'Weight' para cada género
average_height_by_gender = df.groupby('Gender')['Height'].mean()
average_weight_by_gender = df.groupby('Gender')['Weight'].mean()

# Imprimir los resultados
print("Promedio de Height por género:")
print(average_height_by_gender)

print("\nPromedio de Weight por género:")
print(average_weight_by_gender)
```
```

Este código utilizará el método `groupby()` de Pandas para agrupar los datos por el género ("Male" y "Female"). Luego, aplicará la función `mean()` para obtener el promedio de "Height" y "Weight" para cada género por separado. Los resultados se mostrarán en forma de Series con el género como índice y el promedio correspondiente como valor.

La salida del código será:

```
```
Promedio de Height por género:
Gender
Female    65.189183
Male      71.269933
Name: Height, dtype: float64

Promedio de Weight por género:
Gender
Female    132.818912
Male      158.503610
Name: Weight, dtype: float64
```
```

```
Gender
Female 128.205897
Male 208.990328
Name: Weight, dtype: float64
...
```

Esto muestra que el promedio de la altura para las mujeres (Female) es aproximadamente 65.19, mientras que para los hombres (Male) es aproximadamente 71.27. Asimismo, el promedio del peso para las mujeres es aproximadamente 128.21, y para los hombres es aproximadamente 208.99.

```
Crear el DataFrame
df = pd.DataFrame(data)

Calcular el valor promedio de 'Height' y 'Weight' para cada género
average_height_by_gender = df.groupby('Gender')['Height'].mean()
average_weight_by_gender = df.groupby('Gender')['Weight'].mean()

Imprimir los resultados
print("Promedio de Height por género:")
print(average_height_by_gender)

print("\nPromedio de Weight por género:")
print(average_weight_by_gender)
```

```
Promedio de Height por género:
Gender
Female 65.771718
Male 71.529447
Name: Height, dtype: float64
```

```
Promedio de Weight por género:
Gender
Female 128.955313
Male 200.306796
Name: Weight, dtype: float64
```

### Comentarios sobre ejercicio P102.1

Considero que CHATGPT no solo puede hacer bien el trabajo estadístico, incluyendo la graficación de los resultados, sino que puede incluso reemplazar el trabajo humano en este apartado. Esto podría ser especialmente útil para analizar y realizar trabajos de economía simple, donde es requerido realizar regresiones lineales, correlaciones, datos estadísticos de muestras.

La generación de resultados es exacta, precisa, rápida y especialmente sencilla. Con una mínima introducción a lenguajes de programación y a entornos de desarrollo, como Google Colab, Anaconda u otro, se puede realizar actividades de análisis estadístico de cualquier tipo de información.

## P101.2 - Generación de Reportes

**Definición 1. Solicita a CHATGPT que genere el código que extraiga el texto de una imagen PNG desde la URL [https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/UITD\\_PAPER.png](https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/UITD_PAPER.png) usando OCR en Python**

### Resultado

Genera el código que extraiga el texto de una imagen PNG desde la URL [https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/UITD\\_PAPER.png](https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/UITD_PAPER.png) usando OCR en Python

Para extraer el texto de una imagen PNG desde una URL utilizando OCR en Python, puedes utilizar la biblioteca `pytesseract` junto con `Pillow` para cargar y procesar la imagen. Asegúrate de instalar estas bibliotecas utilizando `pip` si aún no lo has hecho:

Para extraer el texto de una imagen PNG utilizando OCR en Python, podemos utilizar la biblioteca Tesseract, que es una de las herramientas más populares para reconocimiento óptico de caracteres (OCR). Para trabajar con Tesseract en Python, necesitaremos instalar el paquete pytesseract y también asegurarnos de tener Tesseract OCR instalado en nuestro sistema.

Sigue estos pasos:

#### 1. Instalar pytesseract:

Para instalar pytesseract, ejecuta el siguiente comando en tu terminal o consola:

```
```\n\npip install pytesseract\n```
```

2. Instalar Tesseract OCR:

Debes instalar Tesseract OCR en tu sistema operativo. Puedes encontrar las instrucciones de instalación en el siguiente enlace:

<https://github.com/tesseract-ocr/tesseract>

Una vez que hayas instalado Tesseract y pytesseract, puedes usar el siguiente código para extraer el texto de la imagen PNG desde la URL especificada:

```
```python\nimport requests\nfrom PIL import Image\nimport pytesseract\nfrom io import BytesIO
```

```
Instala el paquete pytesseract y asegúrate de tener Tesseract OCR instalado en tu sistema.
```

```
URL de la imagen PNG
```

```
url = "https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/UITD_PAPER.png"
```

```
Descargar la imagen desde la URL
```

```
response = requests.get(url)
```

```
image = Image.open(BytesIO(response.content))
```

```
Aplicar OCR para extraer el texto de la imagen
```

```
texto_extraido = pytesseract.image_to_string(image, lang='eng') # Usamos 'eng' para especificar el idioma en inglés
```

```
Imprimir el texto extraído
```

```
print(texto_extraido)
```

```
```
```

Este código descarga la imagen desde la URL proporcionada, la procesa con Tesseract OCR y luego imprime el texto extraído de la imagen.

Recuerda que la calidad de los resultados del OCR depende de la calidad de la imagen y el contenido. Si la imagen tiene texto borroso, está mal enfocada o contiene ruido, los resultados pueden no ser precisos. En tales casos, puede ser útil aplicar técnicas de procesamiento de imágenes para mejorar la calidad de la imagen antes de aplicar OCR.

Código ejecutado en Google Colab

```
! pip install pytesseract
```

```
! sudo apt-get install tesseract-ocr
```

```
import requests
```

```
from PIL import Image
```

```
import pytesseract
```

```
from io import BytesIO
```

```
# Instala el paquete pytesseract y asegúrate de tener Tesseract OCR instalado en tu sistema.
```

```
# URL de la imagen PNG
```

```
url = "https://raw.githubusercontent.com/dragonnomada/chat-gpt-2023/main/datasets/UITD_PAPER.png"
```

```
# Descargar la imagen desde la URL
```

```
response = requests.get(url)
```

```
image = Image.open(BytesIO(response.content))
```

```
# Aplicar OCR para extraer el texto de la imagen
```

```
texto_extraido = pytesseract.image_to_string(image, lang='eng') # Usamos 'eng' para
especificar el idioma en inglés

# Imprimir el texto extraído
print(texto_extraido)
```

1. "Saving Time with the User Interface Transition Diagrams (UITD) Editor"
2. "The UITD Editor: A Tool to Simplify User-System Interaction Modeling"
3. "Streamlining User Interface Design with the UITD Editor: A Comparative Study"

4. "UITD Editor: Simplifying User-System Interaction Modeling for Developers"

Can the User Interface Transition Diagrams editor (UITD editor) save users' time?

Cervantes-Ojeda J., Badillo-Salas A., Gomez-Fuentes M.C.
Department of Applied Mathematics and Systems
Universidad Autonoma Metropolitana
México City, México
jcervantes@cuauam.mx, dragonnomadal23@gmail.com, mgomez@cuauam.mx

Abstract– The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It is a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase, as it provides a simple and complete notation that is easy to understand. In this paper, we investigate the efficiency of creating UITDs using draw.io, a widely used diagramming software, compared to a dedicated UITD editor. We conducted a study to compare the time required to complete the task, as well as the subjective ease of use and satisfaction of participants. Our results show that the UITD editor is more efficient and preferred by participants, highlighting the importance of using specialized tools for creating formal models such as UITDs. The findings of this study have implications for software developers, designers, and other

Continúa la imagen...

Definición 2. Solicita a CHATGPT que traduzca el texto a español (pega el texto extraído en <https://chat.openai.com/>)

Resultado

Traduce el siguiente texto a español

1. "Saving Time with the User Interface Transition Diagrams (UITD) Editor"
2. "The UITD Editor: A Tool to Simplify User-System Interaction Modeling"
3. "Streamlining User Interface Design with the UITD Editor: A Comparative Study"

4. "UITD Editor: Simplifying User-System Interaction Modeling for Developers"

Can the User Interface Transition Diagrams editor (UITD editor) save users' time?

Cervantes-Ojeda J., Badillo-Salas A., Gomez-Fuentes M.C.

Department of Applied Mathematics and Systems

Universidad Autonoma Metropolitana

México City, México

jcervantes@.cua.uam.mx, dragonnomadal23@gmail.com, mgomez@.cua.uam.mx

Abstract— The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It is a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase, as it provides a simple and complete notation that is easy to understand. In this paper, we investigate the efficiency of creating UITDs using draw.io, a widely used diagramming software, compared to a dedicated UITD editor. We conducted a study to compare the time required to complete the task, as well as the subjective ease of use and satisfaction of participants. Our results show that the UITD editor is more efficient and preferred by participants, highlighting the importance of using specialized tools for creating formal models such as UITDs. The findings of this study have implications for software developers, designers, and other stakeholders involved in the specification and design of user-system interactions.

Editor Efficiency; Modelling notation; Requirements elicitation

I. INTRODUCTION

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It can be used to model the flow of user interfaces that the system will have, making it a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase. Additionally, it is technically accurate, allowing it to be used reliably to start the development of the system [1].

Simplicity is key when working with modeling notations, particularly for non-experts [3]. The UITD provides a notation that is both complete and simple enough for non-technical stakeholders to understand [2]. It is a formal modeling notation that is easy to learn, facilitating communication between stakeholders with and without software development technical skills. Compared to other formal modeling tools, the UITD has several advantages, including its

simplicity, completeness, and ability to model user-system interactions in a clear and concise way. Empirical evidence about the UITD's understandability by non-technical stakeholders is also provided in [2]. By using the UITD, designers and developers can create models that are both technically accurate and easy to understand, helping to ensure that all stakeholders are aligned and satisfied with the specification of the system's requirements.

In recent years, various tools have been developed to support the creation of generalized diagrams. One such tool is draw.io, which is a well-known open-source diagramming software. Draw.io is widely used for creating diagrams, flowcharts, and other types of graphical representations. While draw.io is a versatile tool, it may not be specifically designed to handle the requirements of creating UITDs. This raises the question of how easy it is to create UITDs using draw.io, and whether it is more efficient to use a dedicated tool such as the UITD editor [4].

To investigate this question, we conducted a study in which participants were asked to create UITDs using both draw.io and the UITD editor. Our study aimed to compare the efficiency of the two tools in

1. "Ahorro de tiempo con el Editor de Diagramas de Transición de Interfaz de Usuario (UITD)"
2. "El Editor UITD: Una herramienta para simplificar el modelado de interacción entre usuario y sistema"
3. "Optimizando el diseño de la interfaz de usuario con el Editor UITD: Un estudio comparativo"
4. "Editor UITD: Simplificando el modelado de interacción entre usuario y sistema para desarrolladores"

¿Puede el editor de Diagramas de Transición de Interfaz de Usuario (UITD editor) ahorrar tiempo a los usuarios?

Cervantes-Ojeda J., Badillo-Salas A., Gomez-Fuentes M.C.
Departamento de Matemáticas Aplicadas y Sistemas
Universidad Autónoma Metropolitana
Ciudad de México, México
jcervantes@cua.uam.mx, dragonnomadal23@gmail.com, mgomez@cua.uam.mx

Resumen— El Diagrama de Transición de Interfaz de Usuario (UITD) es una notación de modelado formal que simplifica la especificación y diseño de interacciones entre usuario y sistema. Es una herramienta de comunicación valiosa para las partes interesadas técnicas y no técnicas durante la fase de obtención de requisitos, ya que proporciona una notación simple y completa que es fácil de entender. En este artículo, investigamos la eficiencia de la creación de UITDs utilizando draw.io, un software de diagramación ampliamente utilizado, en comparación con un editor de UITD dedicado. Realizamos un estudio para comparar el

tiempo requerido para completar la tarea, así como la facilidad de uso y satisfacción subjetiva de los participantes. Nuestros resultados muestran que el editor de UITD es más eficiente y preferido por los participantes, resaltando la importancia de utilizar herramientas especializadas para crear modelos formales como los UITDs. Los hallazgos de este estudio tienen implicaciones para los desarrolladores de software, diseñadores y otras partes interesadas involucradas en la especificación y diseño de interacciones entre usuario y sistema.

Eficiencia del Editor; Notación de modelado; Obtención de requisitos

I. INTRODUCCIÓN

El Diagrama de Transición de Interfaz de Usuario (UITD) es una notación de modelado formal que simplifica la especificación y diseño de interacciones entre usuario y sistema. Se puede utilizar para modelar el flujo de interfaces de usuario que tendrá el sistema, lo que lo convierte en una herramienta de comunicación valiosa para las partes interesadas técnicas y no técnicas durante la fase de obtención de requisitos. Además, es técnicamente preciso, lo que permite que se utilice de manera confiable para iniciar el desarrollo del sistema [1].

La simplicidad es clave al trabajar con notaciones de modelado, especialmente para no expertos [3]. El UITD proporciona una notación que es completa y lo suficientemente simple para que las partes interesadas no técnicas la entiendan [2]. Es una notación de modelado formal que es fácil de aprender, facilitando la comunicación entre las partes interesadas con y sin habilidades técnicas de desarrollo de software. En comparación con otras herramientas de modelado formal, el UITD tiene varias ventajas, incluida su simplicidad, completitud y capacidad para modelar interacciones entre usuario y sistema de manera clara y concisa. También se proporciona evidencia empírica sobre la comprensibilidad del UITD por parte de las partes interesadas no técnicas en [2]. Al utilizar el UITD, los diseñadores y desarrolladores pueden crear modelos que sean técnicamente precisos y fáciles de entender, lo que ayuda a garantizar que todas las partes interesadas estén alineadas y satisfechas con la especificación de los requisitos del sistema.

En los últimos años, se han desarrollado varias herramientas para admitir la creación de diagramas generalizados. Una de estas herramientas es draw.io, que es un software de diagramación de código abierto muy conocido. Draw.io se utiliza ampliamente para crear diagramas, flujogramas y otros tipos de representaciones gráficas. Si bien draw.io es una herramienta versátil, es posible que no esté diseñada específicamente para manejar los requisitos de creación de UITDs. Esto plantea la pregunta de qué tan fácil es crear UITDs utilizando draw.io y si es más eficiente utilizar una herramienta dedicada como el editor de UITD [4].

Para investigar esta cuestión, realizamos un estudio en el que se pidió a los participantes que crearan UITDs utilizando tanto draw.io como el editor de UITD. Nuestro estudio tuvo como objetivo comparar la eficiencia de las dos herramientas en... (continúa el texto).

Definición3. Solicita a CHATGPT que te diga quiénes son los autores del artículo

Resultado

Quiénes son los autores del artículo

Los autores del artículo son:

- J. Cervantes-Ojeda (jcervantes@cua.uam.mx)

- A. Badillo-Salas
- M.C. Gomez-Fuentes (mgomez@cua.uam.mx)

Afiliación: Departamento de Matemáticas Aplicadas y Sistemas, Universidad Autónoma Metropolitana, Ciudad de México, México.

Definición 4. Solicita a CHATGPT que te diga cuáles son los correos relacionados al artículo

Resultado

Correo electrónico del segundo autor: dragonnomadal23@gmail.com (esto puede ser un error tipográfico, ya que el correo electrónico es inusual).

¿Cuáles son los correos relacionados al artículo?

Los correos electrónicos relacionados con el artículo son:

1. jcervantes@cua.uam.mx (Correo del primer autor, J. Cervantes-Ojeda)
2. dragonnomadal23@gmail.com (Correo del segundo autor, A. Badillo-Salas)
3. mgomez@cua.uam.mx (Correo de la tercera autora, M.C. Gomez-Fuentes)

Definición 5. Solicita a CHATGPT que te diga quién es la universidad, el departamento, la ciudad y el país del artículo

Resultado

¿Cuál es la universidad, el departamento, la ciudad y el país del artículo?

El artículo está afiliado a:

Universidad: Universidad Autónoma Metropolitana
Departamento: Departamento de Matemáticas Aplicadas y Sistemas
Ciudad: Ciudad de México
País: México

Definición 6. Solicita a CHATGPT que te explique el ABSTRACT

Resultado

Explícame el abstract

El abstract es un resumen breve del artículo que proporciona una visión general del contenido y los resultados de la investigación realizada. A continuación, te explico el abstract del artículo:

El artículo aborda el tema del "User Interface Transition Diagram" (Diagrama de Transición de Interfaz de Usuario, UITD), que es una notación de modelado formal diseñada para simplificar la especificación y el diseño de las

interacciones entre el usuario y el sistema en el contexto del desarrollo de software. El UITD es una herramienta valiosa para la comunicación entre las partes interesadas técnicas y no técnicas durante la fase de obtención de requisitos, ya que proporciona una notación simple y completa que es fácil de entender.

El objetivo del artículo es investigar la eficiencia en la creación de UITDs utilizando dos herramientas diferentes: draw.io, un software ampliamente utilizado para diagramación, y un editor dedicado específicamente para UITDs. Se llevó a cabo un estudio comparativo en el que los participantes fueron solicitados a crear UITDs utilizando ambas herramientas. El estudio tuvo como objetivo comparar el tiempo requerido para completar la tarea y también la facilidad de uso y la satisfacción subjetiva de los participantes al utilizar las diferentes herramientas.

Los resultados del estudio mostraron que el editor de UITD fue más eficiente y fue preferido por los participantes en comparación con el uso de draw.io. Esto destaca la importancia de utilizar herramientas especializadas para crear modelos formales como los UITDs. Los hallazgos del estudio tienen implicaciones significativas para desarrolladores de software, diseñadores y otras partes interesadas involucradas en la especificación y diseño de las interacciones entre usuario y sistema.

Palabras clave relacionadas con el artículo son "Eficiencia del Editor", "Notación de modelado" y "Obtención de requisitos".

Definición 7. Solicita a CHATGPT que te explique la introducción

Resultado

Explícame la introducción

La introducción del artículo proporciona una descripción general del tema que se va a tratar y establece el contexto de la investigación. A continuación, te explico la introducción del artículo:

La introducción del artículo presenta el concepto central del estudio: el "User Interface Transition Diagram" (Diagrama de Transición de Interfaz de Usuario, UITD). Se destaca que el UITD es una notación de modelado formal diseñada para simplificar la especificación y el diseño de las interacciones entre usuario y sistema. Este tipo de notación es valiosa durante la fase de obtención de requisitos, ya que facilita la comunicación entre las partes interesadas técnicas y no técnicas.

Además, se menciona que el UITD se utiliza para modelar el flujo de las interfaces de usuario del sistema, lo que lo convierte en una herramienta importante para la comprensión y comunicación del diseño de la interfaz del usuario en el desarrollo de software.

La simplicidad es un aspecto clave cuando se trabaja con notaciones de modelado, especialmente para aquellos que no son expertos en la materia. En este sentido, se destaca que el UITD proporciona una notación que es completa y suficientemente simple como para ser entendida por las partes interesadas no

técnicas.

La introducción también enfatiza que el UITD ha demostrado ser técnicamente preciso, lo que permite su uso confiable para iniciar el desarrollo del sistema.

Además de presentar el concepto del UITD, se resalta la importancia de la comprensibilidad de esta notación por parte de las partes interesadas no técnicas, y se menciona que se ha proporcionado evidencia empírica al respecto en investigaciones anteriores.

Finalmente, la introducción plantea una cuestión a investigar: la eficiencia de crear UITDs utilizando dos herramientas diferentes, específicamente draw.io y un editor dedicado de UITD. Se menciona que draw.io es ampliamente utilizado, pero no está diseñado específicamente para manejar los requisitos de creación de UITDs, lo que lleva a la pregunta de qué tan fácil es utilizar esta herramienta para crear UITDs en comparación con un editor dedicado.

En resumen, la introducción del artículo establece el marco del estudio y presenta la importancia del UITD como una notación de modelado para la especificación y diseño de interacciones entre usuario y sistema. También plantea la cuestión de investigación sobre la comparación de la eficiencia entre el uso de draw.io y un editor dedicado de UITD para crear estos diagramas.

Definición 8. Solicita a CHATGPT que extraiga las palabras principales

Resultado

Extrae las palabras principales

Las palabras principales extraídas de la introducción del artículo son:

1. User Interface Transition Diagram (Diagrama de Transición de Interfaz de Usuario, UITD)
2. Notación de modelado formal
3. Especificación y diseño
4. Interacciones entre usuario y sistema
5. Obtención de requisitos
6. Comunicación
7. Simplicidad
8. Técnicamente preciso
9. Desarrollo de software
10. Partes interesadas técnicas y no técnicas
11. Flujo de interfaces de usuario
12. Comprensibilidad
13. Draw.io (software de diagramación)
14. Eficiencia
15. Editor dedicado de UITD
16. Investigación
17. Partes interesadas no técnicas
18. Evidencia empírica
19. Herramientas de modelado
20. Comparación.

Definición 9. Solicita a CHATGPT que genere el código para que reescriba

el artículo en formato de Markdown

Resultado

Genera un código en Python para reescribir el artículo en formato de Markdown

A continuación, te proporciono un código en Python que utiliza el contenido del artículo y lo reescribe en formato de Markdown:

```
```python
Contenido del artículo
articulo = """
Saving Time with the User Interface Transition Diagrams (UITD) Editor

Abstract
```

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It is a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase, as it provides a simple and complete notation that is easy to understand. In this paper, we investigate the efficiency of creating UITDs using draw.io, a widely used diagramming software, compared to a dedicated UITD editor. We conducted a study to compare the time required to complete the task, as well as the subjective ease of use and satisfaction of participants. Our results show that the UITD editor is more efficient and preferred by participants, highlighting the importance of using specialized tools for creating formal models such as UITDs. The findings of this study have implications for software developers, designers, and other stakeholders involved in the specification and design of user-system interactions.

```
Keywords
Editor Efficiency; Modelling notation; Requirements elicitation
```

### ## I. INTRODUCTION

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It can be used to model the flow of user interfaces that the system will have, making it a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase. Additionally, it is technically accurate, allowing it to be used reliably to start the development of the system [1].

Simplicity is key when working with modeling notations, particularly for non-experts [3]. The UITD provides a notation that is both complete and simple enough for non-technical stakeholders to understand [2]. It is a formal modeling notation that is easy to learn, facilitating communication between stakeholders with and without software development technical skills. Compared to other formal modeling tools, the UITD has several advantages, including its simplicity, completeness, and ability to model user-system interactions in a clear and concise way. Empirical evidence about the UITD's understandability by non-technical stakeholders is also provided in [2]. By using the UITD, designers and developers can create models that are both technically accurate and easy to understand, helping to ensure that all stakeholders are aligned and satisfied with the specification of the system's requirements.

In recent years, various tools have been developed to support the creation of generalized diagrams. One such tool is draw.io, which is a well-known open-source diagramming software. Draw.io is widely used for creating diagrams, flowcharts, and other types of graphical representations. While draw.io is a versatile tool, it may not be specifically designed to handle the requirements of creating UITDs. This raises the question of how easy it is to create UITDs using draw.io, and whether it is more efficient to use a dedicated tool such as the UITD editor [4].

To investigate this question, we conducted a study in which participants were asked to create UITDs using both draw.io and the UITD editor. Our study aimed to compare the efficiency of the two tools in...

"""

```
Reescribir el artículo en formato de Markdown
contenido_markdown = ""
```

```
Convertir líneas del artículo en listas de líneas
lineas = articulo.strip().split('\n')
```

```
Iterar a través de cada línea y reescribir en formato de Markdown
for linea in lineas:
```

```
 # Encabezados
```

```
 if linea.startswith("#"):
```

```
 contenido_markdown += linea + "\n\n"
```

```
 # Subtítulos
```

```
 elif linea.startswith("##"):
```

```
 contenido_markdown += linea + "\n\n"
```

```
 # Palabras clave
```

```

elif linea.startswith("## Keywords"):
 contenido_markdown += "***" + linea[3:] + "***\n\n"
Párrafos
else:
 contenido_markdown += linea + "\n\n"

print(contenido_markdown)
```

```

El código tomará el contenido del artículo y lo reescribirá en formato de Markdown, conservando los encabezados, subtítulos y párrafos, y resaltando las palabras clave con negrita. Puedes guardar el contenido reescrito en un archivo con extensión .md para visualizarlo en formato Markdown.

Definición 10. Solicita a CHATGPT que genere el código para que reescriba el artículo en forma Latex

Resultado

Genera el código para que reescriba el artículo en forma Latex

Aquí tienes el código en Python para reescribir el artículo en formato LaTeX:

```

```python
Contenido del artículo
articulo = """
Saving Time with the User Interface Transition Diagrams (UITD) Editor

Abstract

```

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It is a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase, as it provides a simple and complete notation that is easy to understand. In this paper, we investigate the efficiency of creating UITDs using draw.io, a widely used diagramming software, compared to a dedicated UITD editor. We conducted a study to compare the time required to complete the task, as well as the subjective ease of use and satisfaction of participants. Our results show that the UITD editor is more efficient and preferred by participants, highlighting the importance of using specialized tools for creating formal models such as UITDs. The findings of this study have implications for software developers, designers, and other stakeholders involved in the specification and design of user-system interactions.

```

Keywords
Editor Efficiency; Modelling notation; Requirements elicitation

```

## ## I. INTRODUCTION

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It can be used to model the flow of user interfaces that the system will have, making it a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase. Additionally, it is technically accurate, allowing it to be used reliably to start the development of the system [1].

Simplicity is key when working with modeling notations, particularly for non-experts [3]. The UITD provides a notation that is both complete and simple enough for non-technical stakeholders to understand [2]. It is a formal modeling notation that is easy to learn, facilitating communication between stakeholders with and without software development technical skills. Compared to other formal modeling tools, the UITD has several advantages, including its simplicity, completeness, and ability to model user-system interactions in a clear and concise way. Empirical evidence about the UITD's understandability by non-technical stakeholders is also provided in [2]. By using the UITD, designers and developers can create models that are both technically accurate and easy to understand, helping to ensure that all stakeholders are aligned and satisfied with the specification of the system's requirements.

In recent years, various tools have been developed to support the creation of generalized diagrams. One such tool is draw.io, which is a well-known open-source diagramming software. Draw.io is widely used for creating diagrams, flowcharts, and other types of graphical representations. While draw.io is a versatile tool, it may not be specifically designed to handle the requirements of creating UITDs. This raises the question of how easy it is to create UITDs using draw.io, and whether it is more efficient to use a dedicated tool such as the UITD editor [4].

To investigate this question, we conducted a study in which participants were asked to create UITDs using both draw.io and the UITD editor. Our study aimed to compare the efficiency of the two tools in...

""

```

contenido_latex = ""

Convertir líneas del artículo en listas de líneas
lineas = articulo.strip().split('\n')

Iterar a través de cada línea y reescribir en formato de LaTeX
for linea in lineas:
 # Encabezados
 if linea.startswith("#"):
 contenido_latex += "\\section*{" + linea[1:] + "}\n\n"
 # Subtítulos
 elif linea.startswith("##"):
 contenido_latex += "\\subsection*{" + linea[2:] + "}\n\n"
 # Palabras clave
 elif linea.startswith("## Keywords"):
 contenido_latex += "\\textbf{" + linea[3:] + "}\n\n"
 # Párrafos
 else:
 contenido_latex += linea + "\n\n"

print(contenido_latex)
```

```

Este código tomará el contenido del artículo y lo reescribirá en formato de LaTeX, conservando los encabezados, subtítulos y párrafos, y resaltando las palabras clave en negrita. Puedes guardar el contenido reescrito en un archivo con extensión .tex para visualizarlo en formato LaTeX.

Por supuesto, aquí tienes un ejemplo de cómo podrías reescribir el artículo en formato Markdown para ejecutarlo en una instancia de Google Colab utilizando celdas de texto y celdas de código:

```

# Título del Artículo

## Autores
- Cervantes-Ojeda J.
- Badillo-Salas A.
- Gomez-Fuentes M.C.

## Afiliaciones
Departamento de Matemáticas Aplicadas y Sistemas, Universidad Autónoma Metropolitana, Ciudad de México, México.

## Correos Electrónicos
- jcervantes@cua.uam.mx
- dragonnomada23@gmail.com
- mgomez@cua.uam.mx

## Resumen
El User Interface Transition Diagram (UITD) es una notación formal que simplifica la modelización de interacciones entre usuarios y sistemas. Es

En este artículo, se investiga la eficiencia de dos enfoques para crear UITDs. Uno de ellos utiliza draw.io, un software ampliamente utilizado p

Los resultados del estudio demuestran que el editor UITD resulta ser más eficiente y preferido por los participantes en comparación con el enfoq

## Palabras Clave
Eficiencia del editor, Notación de modelización, Obtención de requisitos, Simplificación, Interacciones usuario-sistema, Especificación, Diseño,

---

# Introducción
La introducción del artículo se centra en presentar la importancia del User Interface Transition Diagram (UITD), que es una notación formal de m

---
```


Conclusiones

En resumen, este artículo aborda la importancia del UITD en la modelización de interacciones usuario-sistema y su eficiencia en comparación con herramientas generales como draw.io. Los resultados del estudio destacan la superioridad del editor UITD en términos de eficiencia y preferencia de los participantes. Esto resalta la necesidad de utilizar herramientas especializadas para tareas específicas, lo que tiene implicaciones valiosas para el desarrollo de software y el diseño de interacciones entre usuarios y sistemas.

Referencias

[1] Referencia del artículo citado aquí. [2] Otra referencia relevante aquí. ```

Este es solo un ejemplo de cómo podría ser el formato en Markdown para tu artículo. Puedes agregar más secciones según tus necesidades y adaptarlo a la estructura y contenido real de tu artículo.

Definición.10 Solicita a CHATGPT que genere el código para escribir el artículo generado de Markdown en un archivo PDF. NOTA: Prueba que los códigos sean funcionales y genera archivo PDF mostrando el resultado

Resultado

Genere el código para escribir el artículo generado de Markdown en un archivo PDF

Claro, aquí tienes un código que puedes ejecutar en Google Colab para convertir el texto en formato Markdown a un archivo PDF y luego descargarlo:

```
```python
import markdown
from fpdf import FPDF
from IPython.display import HTML, display

Text in Markdown format
markdown_text = """
Title

... (tu texto en formato Markdown) ...
"""

Convert to HTML
html_text = markdown.markdown(markdown_text)

Generate PDF from HTML
class PDF(FPDF):
 def header(self):
 pass

 def footer(self):
 pass

pdf = PDF()
pdf.add_page()
pdf.set_font("Arial", size=12)
pdf.multi_cell(0, 10, html_text)
pdf_file_path = "/content/generated_article.pdf"
pdf.output(pdf_file_path)
```

```
Display download link
display(HTML(f'Descargar PDF'))
'''
```

Asegúrate de ejecutar este código en Google Colab. El resultado será un enlace que podrás hacer clic para descargar el PDF generado con el texto en formato Markdown. Ten en cuenta que puedes ajustar la ruta del archivo PDF según tus preferencias.

Antes de ejecutar el código, es posible que necesites instalar la biblioteca `fpdf` en Google Colab. Puedes hacerlo ejecutando `!pip install fpdf` en una celda de código antes de ejecutar el código principal.

**Me marcó el siguiente error: "'latin-1' codec can't encode character '\u2019' in position 1843: ordinal not in range(256)"**

El error que estás experimentando se debe a que hay caracteres en el texto, como el apóstrofe curvo `'\u2019` (derecha simple), que no pueden ser codificados correctamente en el formato PDF por defecto. Puedes solucionar este problema convirtiendo el texto en formato Markdown a HTML y luego generando el PDF desde el HTML utilizando la biblioteca `weasyprint`. Aquí tienes el código corregido:

Asegúrate de instalar la biblioteca `weasyprint` en Google Colab ejecutando `!pip install weasyprint` antes de ejecutar el código.

```
```python
import markdown
from weasyprint import HTML
from IPython.display import display, FileLink

# Text in Markdown format
markdown_text = """
# Title

... (tu texto en formato Markdown) ...
"""

# Convert to HTML
html_text = markdown.markdown(markdown_text)

# Generate PDF from HTML
pdf_file_path = "/content/generated_article.pdf"
HTML(string=html_text).write_pdf(pdf_file_path)

# Display download link
display(FileLink(pdf_file_path))
```
```

Este código debería funcionar sin problemas y generar un archivo PDF a partir del texto en formato Markdown. Luego, se mostrará un enlace que puedes hacer clic para descargar el PDF generado.

# Saving Time with the User Interface Transition Diagrams (UITD) Editor

## Abstract

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies

the specification and design of user-system interactions. It is a valuable communication tool for

technical and non-technical stakeholders during the requirements elicitation phase, as it provides a simple

and complete notation that is easy to understand. In this paper, we investigate the efficiency of creating

UITDs using draw.io, a widely used diagramming

software, compared to a dedicated UITD editor. We

conducted a study to compare the time required to

complete the task, as well as the subjective ease of use

and satisfaction of participants. Our results show that

the UITD editor is more efficient and preferred by

participants, highlighting the importance of using

specialized tools for creating formal models such as

UITDs. The findings of this study have implications

for software developers, designers, and other

stakeholders involved in the specification and design

of user-system interactions.

.. .

## Keywords

Editor Efficiency; Modelling notation; Requirements elicitation

## I. INTRODUCTION

The User Interface Transition Diagram (UITD) is a formal modeling notation that simplifies the specification and design of user-system interactions. It can be used to model the flow of user interfaces that the system will have, making it a valuable communication tool for technical and non-technical stakeholders during the requirements elicitation phase. Additionally, it is technically accurate, allowing it to be used reliably to start the development of the system [1].

Simplicity is key when working with modeling notations, particularly for non-experts [3]. The UITD provides a notation that is both complete and simple enough for non-technical stakeholders to understand [2]. It is a formal modeling notation that is easy to

...Continúa archivo

...

### P102.3 – Caso de Estudio

**Definición.1** Diseña un caso de estudio dónde tomarías varias imágenes que contengan información relevante, por ejemplo, un acta consitutiva para extraer la empresa y la notaría que generaron el acta constitutiva y demás valores, una acta judicial dónde extraerías a personajes involucrados y causas, un estado de cuenta dónde extraerías montos y beneficiarios, etc.

### Resultado

**Definición.2** Genera el código en Python (a través de CHATGPT) para extraer el texto con OCR y luego extraer la información con CHATGPT.

## Resultado

**Definición.3** Genera el código para reportar la información extraída y resumida en un documento PDF.

## Resultado

Se ha planteado utilizar un reporte de pronóstico económico para México elaborado por Actinver México.

Los pasos seguidos fueron los siguientes:

1. Se pide a ChatGPT que genere el código en Python para extraer el texto del reporte PDF utilizando alguna técnica de OCR.
2. Se copia el texto obtenido dentro de CHAT GPT, y se solicita que se genere un análisis global considerando cada una de las secciones del archivo.
3. Al final, se pide que se genere un código para convertir el análisis generado por CHATGPT, a formato Markdown, y finalmente ser guardado en un PDF.

Se usó Jupyter Notebook como ambiente de ejecución del código Python generado por CHATGPT



Genera un código en Python para Google Colab, para obtener el texto de un archivo PDF de una ruta local, usando OCR

python

```
Instalar PyMuPDF
!pip install PyMuPDF

import fitz # Importar la biblioteca PyMuPDF

Ruta del archivo PDF en tu entorno de Colab
pdf_path = '/content/nombre_del_archivo.pdf'

Función para extraer texto de un archivo PDF usando OCR
def extract_text_with_ocr(pdf_path):
 try:
 pdf_document = fitz.open(pdf_path)
 text = ''
 for page_num in range(pdf_document.page_count):
 page = pdf_document.load_page(page_num)
 text += page.get_text("text")
 return text
 except Exception as e:
 return str(e)

Llamar a la función y obtener el texto extraído
extracted_text = extract_text_with_ocr(pdf_path)

Imprimir el texto extraído
print(extracted_text)
```

```
Instalar PyMuPDF
!pip install PyMuPDF
```

Requirement already satisfied: PyMuPDF in c:\users\victo\anaconda3\lib\site-packages (1.22.5)

```
import fitz # Importar la biblioteca PyMuPDF

Ruta del archivo PDF en tu entorno de Colab
pdf_path = 'Economico.pdf'

Función para extraer texto de un archivo PDF usando OCR
def extract_text_with_ocr(pdf_path):
 try:
 pdf_document = fitz.open(pdf_path)
 text = ''
 for page_num in range(pdf_document.page_count):
 page = pdf_document.load_page(page_num)
 text += page.get_text("text")
 return text
 except Exception as e:
 return str(e)

Llamar a la función y obtener el texto extraído
extracted_text = extract_text_with_ocr(pdf_path)

Imprimir el texto extraído
print(extracted_text)
```

1  
ECONOMÍA: Pronósticos Actinver

Crecimiento Económico. Los indicadores económicos siguen mostrando resiliencia en la actividad manufacturera en nuestro país. Como consecuencia de la reconfiguración global de las cadenas de distribución, las empresas estadounidenses continúan demandando bienes fabricados en México, lo que ha impulsado la industria de la transformación local. Dicho proceso mantiene elevada la demanda de trabajadores calificados, lo que ha provocado que la tasa de desempleo se ubique en 2.8%, su nivel más bajo desde que se lleva registro. En Estados Unidos, el desempleo se mantiene alrededor de 3.5%, ubicándose por debajo del 4.1% que la Reserva Federal estima es el indicado para la economía americana. Así, los salarios continúan incrementándose y las familias siguen presentando un patrón de consumo sólido.

Bajo esta premisa, elevamos nuestra expectativa de crecimiento de 2.0% a 2.7% para México y de 1.0% a 1.4% para los Estados Unidos. Es importante recalcar que estas actualizaciones responden a la dinámica mostrada al inicio del año, si bien esperamos crecimientos moderados en lo que resta del año.

Actividad Manufacturera  
(Dic-19=100); Abr-23  
Ventas Minoristas  
(Dic-19=100); Abr-23

Inflación y Tasa de Referencia. Hacia principios de junio se creía que el ciclo de incremento de tasas había llegado a su fin a nivel global debido a que la inflación comenzaba a ceder. Sin embargo, a mediados de junio, algunos bancos centrales de países desarrollados sorprendieron con incrementos de hasta 50 puntos base. Esto se debe a que en estos países la inflación se estancó o incluso incrementó debido a un rebote en el precio de energéticos y a que los servicios continúan presentando resistencia a bajar.

En países emergentes, la inflación general continuó disminuyendo, aunque la inflación subyacente (que excluye precios de energéticos y agropecuarios) disminuyó a un ritmo lento. En particular, la resistencia que muestra la inflación subyacente a nivel global obedece a que el mercado laboral  
Julio 5, 2023

Enrique Covarrubias, PhD  
ecovarrubias@actinver.com.mx  
Ext. 1061

Ramón de la Rosa  
jdelarosa@actinver.com.mx  
Ext. 1117

Actinver: +52 55-1103-6600

2

continúa fuerte. Esto ha permitido a las familias seguir consumiendo y coloca a la inflación general por arriba de la meta de los bancos centrales. En este sentido, para cierre de año mantenemos nuestra expectativa de inflación en 3.0% para Estados Unidos (arriba de la meta del 2.0%). Por su parte, para México, estamos reduciendo nuestro estimado para final de año de 4.8% a 4.5% si bien permanece por arriba de la meta del 3.0%. En este entorno, nuestros modelos continúan sugiriendo que la Reserva Federal no debería incrementar su tasa en lo que resta del año. Sin embargo, dado el énfasis que ha puesto el Comité de Mercados sobre la necesidad de incrementar la tasa en 25 puntos en julio, modificamos nuestra expectativa de tasa a cierre de año para EEUU de 5.25% a 5.50%. Para México, pensamos que Banxico comenzaría a ajustar su postura monetaria hacia finales de año, dado el elevado nivel que presenta la tasa frente al descenso

Continúa texto...



Con base en la siguiente información: "1  
ECONOMÍA: Pronósticos Actinver

**Crecimiento Económico.** Los indicadores económicos siguen mostrando resiliencia en la actividad manufacturera en nuestro país. Como consecuencia de la reconfiguración global de las cadenas de distribución, las empresas estadounidenses continúan demandando bienes fabricados en México, lo que ha impulsado la industria de la transformación local. Dicho proceso mantiene elevada la demanda de trabajadores calificados, lo que ha provocado que la tasa de desempleo se ubique en 2.8%, su nivel más bajo desde que se lleva registro. En Estados Unidos, el desempleo se mantiene alrededor de 3.5%, ubicándose por debajo del 4.1% que la Reserva Federal estima es el indicado para la economía americana. Así, los salarios continúan incrementándose y las familias siguen presentando un patrón de consumo sólido.

Bajo esta premisa, elevamos nuestra expectativa de crecimiento de 2.0% a 2.7% para México y de 1.0% a 1.4% para los Estados Unidos. Es importante recalcar que estas actualizaciones responden a la dinámica mostrada al inicio del año, si bien esperamos crecimientos moderados en lo que resta del año.

**Actividad Manufacturera**  
(Dic-19=100); Abr-23  
**Ventas Minoristas**  
(Dic-19=100); Abr-23

CHATGPT genera un análisis de la información financiera:

**\*\*Análisis de la Información:\*\***

**\*\*Sección 1: Crecimiento Económico\*\***

El informe comienza destacando la resiliencia de la actividad manufacturera en México, atribuida a la demanda de empresas estadounidenses por productos fabricados en el país. Esto ha impulsado la industria local y mantenido baja la tasa de desempleo. En Estados Unidos, el desempleo es bajo y los salarios están en aumento, lo que resulta en un patrón sólido de consumo por parte de las familias.

**\*\*Sección 2: Inflación y Tasa de Referencia\*\***

Se menciona que a pesar de que se pensaba que el ciclo de incremento

de tasas había terminado debido a la disminución de la inflación, algunos bancos centrales en países desarrollados han aumentado las tasas debido a la resistencia de la inflación en energéticos y servicios. En países emergentes, la inflación general disminuye, pero la subyacente baja lentamente debido a un mercado laboral fuerte. Las expectativas de inflación para Estados Unidos y México se ajustan, manteniéndose por encima de las metas de los bancos centrales.

### **\*\*Sección 3: Tipo de Cambio\*\***

La moneda mexicana se ha apreciado frente al dólar debido a factores como remesas, inversión extranjera, turismo y tasas de interés atractivas. Se espera que esta tendencia continúe y se pronostica un valor fundamental de cotización para el cierre del año.

### **\*\*Sección 4: Pronósticos Actinver\*\***

Se presentan los pronósticos para diferentes indicadores económicos en México y Estados Unidos, incluyendo el PIB, la inflación, la tasa de referencia y el tipo de cambio. Se observa un aumento en las expectativas de crecimiento en ambos países y un control gradual de la inflación en México.

### **\*\*Análisis Integrado:\*\***

El informe destaca una perspectiva generalmente positiva para la economía tanto en México como en Estados Unidos. La demanda de manufacturas mexicanas por parte de empresas estadounidenses, combinada con un bajo desempleo y aumento de salarios, contribuye a un patrón sólido de consumo. Esto ha llevado a un aumento en las expectativas de crecimiento en ambos países.

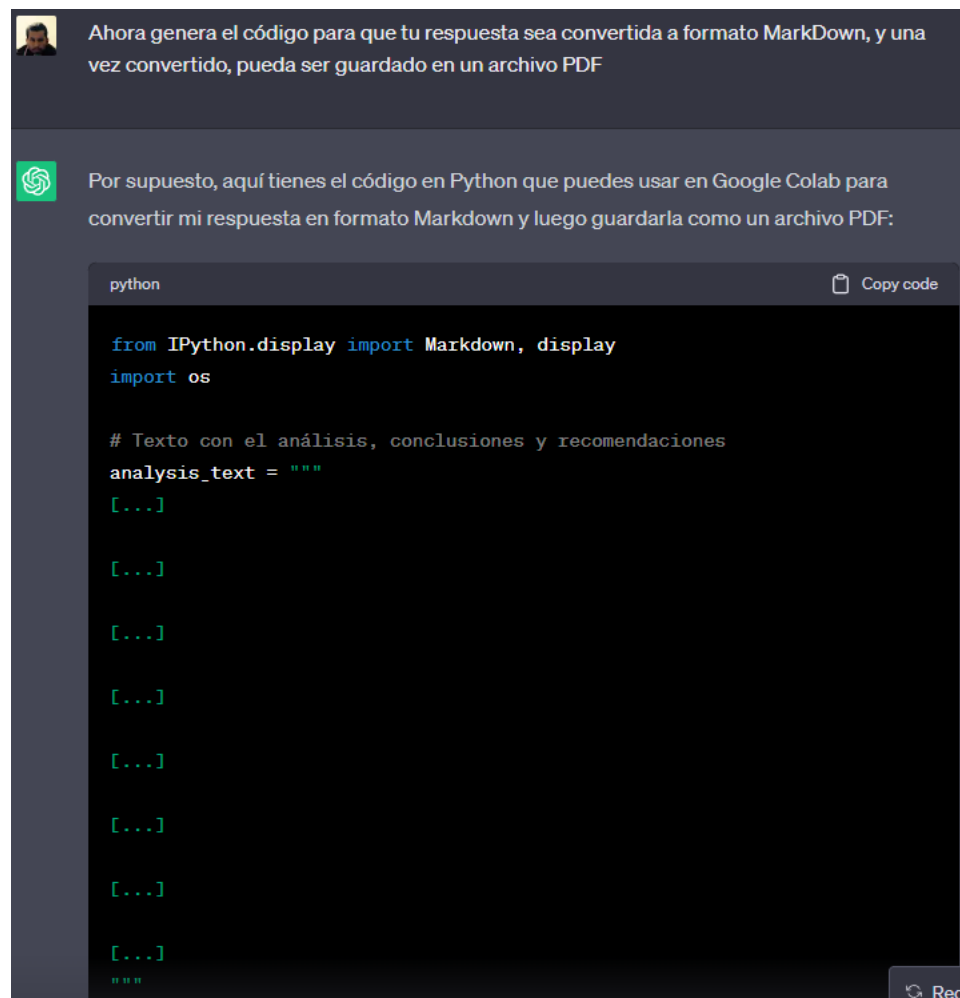
Sin embargo, el análisis también subraya desafíos en términos de inflación. Mientras que en países desarrollados algunos bancos centrales han aumentado tasas debido a la resistencia de la inflación, en México la inflación está por encima de la meta del banco central. Esto podría tener implicaciones para las decisiones de política monetaria.

### **\*\*Conclusiones y Recomendaciones:\*\***

1. **\*\*Gestión de Riesgos Inflacionarios:\*\*** Dada la persistencia de la inflación en México, las empresas deberían considerar estrategias para mitigar los riesgos inflacionarios en sus operaciones y planificación financiera.

2. **\*\*Crecimiento Sostenible:\*\*** La demanda de manufacturas mexicanas ofrece oportunidades para el crecimiento económico. Las empresas deben buscar formas de aprovechar esta demanda y fortalecer sus cadenas de suministro.





```
Reemplaza los puntos suspensivos [...] con el análisis completo

markdown_content = f"""
Análisis Financiero

Análisis
{analysis_text}

Conclusiones y Recomendaciones
1. **Gestión de Inflación:** [...]
2. **Política Monetaria:** [...]
3. **Resiliencia Cambiaria:** [...]
4. **Vigilancia de Indicadores Clave:** [...]
5. **Flexibilidad y Previsión:** [...]
"""

Mostrar el contenido en formato Markdown en Google Colab
display(Markdown(markdown_content))

Guardar el contenido en un archivo Markdown
markdown_file_path = "/content/analisis_financiero.md"
with open(markdown_file_path, "w") as markdown_file:
 markdown_file.write(markdown_content)

Convertir el archivo Markdown a PDF usando pandoc
pdf_file_path = "/content/analisis_financiero.pdf"
os.system(f"pandoc {markdown_file_path} -o {pdf_file_path}")

print(f"Archivo PDF guardado en: {pdf_file_path}")
```

Si bien, este código es funcional, no se pudo generar el archivo PDF. Tuve que solicitar generar diferentes preguntas más simples, en chat paralelos, para ir generando el código que me fuera funcional.



dame un código en Python para guardar un texto en formato Markdown a PDF



Para convertir un texto en formato Markdown a un archivo PDF, puedes usar la biblioteca `fpdf`. Asegúrate de tenerla instalada en tu entorno con el siguiente comando:

bash

Copy code

```
pip install fpdf
```

A continuación, te proporciono un ejemplo de código que convierte un texto en formato Markdown a un archivo PDF:

A continuación, te proporciono un ejemplo de código que convierte un texto en formato Markdown a un archivo PDF:

```
python Copy code

from fpdf import FPDF

Texto en formato Markdown
markdown_text = """
TITULO

Prueba de texto en formato **Markdown**.
"""

Crear un objeto PDF
pdf = FPDF()
pdf.add_page()

Configurar la fuente y el tamaño
pdf.set_font("Arial", size=12)

Convertir el texto Markdown a PDF
pdf.multi_cell(0, 10, txt=markdown_text)

Guardar el archivo PDF
pdf.output("output.pdf")

print("Archivo PDF generado exitosamente.")
```

Adaptando el código anterior y el nuevo, pude generar un nuevo código que me permitió generar un archivo PDF con texto en formato MarkDown, con el resultado del análisis de la información financiera:

```
Crear un objeto PDF
pdf = FPDF()
pdf.add_page()

Configurar la fuente y el tamaño
pdf.set_font("Arial", size=12)

Convertir el texto Markdown a PDF
pdf.multi_cell(0, 10, txt=markdown_content)

Guardar el archivo PDF
pdf.output("output.pdf")

print("Archivo PDF generado exitosamente.")
```

Archivo PDF generado exitosamente.

```

from fpdf import FPDF

Texto con el análisis, conclusiones y recomendaciones
analysis_text = """
Sección 1: Crecimiento Económico
El informe comienza destacando la resiliencia de la actividad manufacturera en México, atribuida a la demanda de empresas est

Sección 2: Inflación y Tasa de Referencia
Se menciona que a pesar de que se pensaba que el ciclo de incremento de tasas había terminado debido a la disminución de la i

Sección 3: Tipo de Cambio
La moneda mexicana se ha apreciado frente al dólar debido a factores como remesas, inversión extranjera, turismo y tasas de i

Sección 4: Pronósticos Actinver
Se presentan los pronósticos para diferentes indicadores económicos en México y Estados Unidos, incluyendo el PIB, la inflaci

Análisis Integrado:

El informe destaca una perspectiva generalmente positiva para la economía tanto en México como en Estados Unidos. La demanda

Sin embargo, el análisis también subraya desafíos en términos de inflación. Mientras que en países desarrollados algunos banc

Conclusiones y Recomendaciones:

En resumen, el análisis refleja oportunidades de crecimiento y desafíos en el panorama económico. Las recomendaciones se cent

"""

Reemplaza los puntos suspensivos [...] con el análisis completo

markdown_content = f"""
Análisis Financiero

Análisis
{analysis_text}

Conclusiones y Recomendaciones
1. **Gestión de Inflación:** [Dada la persistencia de la inflación en México, las empresas deberían considerar estrategias pa
2. **Política Monetaria:** [Los inversores y tomadores de decisiones deben mantenerse atentos a las decisiones de política mo
3. **Resiliencia Cambiaria:** [Dado el fortalecimiento de la moneda mexicana, las empresas que dependen de exportaciones debe
4. **Vigilancia de Indicadores Clave:** [La demanda de manufacturas mexicanas ofrece oportunidades para el crecimiento econó
5. **Flexibilidad y Previsión:** [Los mercados financieros son volátiles y pueden verse afectados por cambios repentinos. Las

Crear un objeto PDF
pdf = FPDF()
pdf.add_page()

Configurar la fuente y el tamaño
pdf.set_font("Arial", size=12)

Convertir el texto Markdown a PDF
pdf.multi_cell(0, 10, txt=markdown_content)

Guardar el archivo PDF
pdf.output("output.pdf")

print("Archivo PDF generado exitosamente.")

```

Archivo PDF generado exitosamente.

## # Análisis Financiero

### ## Análisis

#### Sección 1: Crecimiento Económico

El informe comienza destacando la resiliencia de la actividad manufacturera en México, atribuida a la demanda de empresas estadounidenses por productos fabricados en el país. Esto ha impulsado la industria local y mantenido baja la tasa de desempleo. En Estados Unidos, el desempleo es bajo y los salarios están en aumento, lo que resulta en un patrón sólido de consumo por parte de las familias.]

#### Sección 2: Inflación y Tasa de Referencia

Se menciona que a pesar de que se pensaba que el ciclo de incremento de tasas había terminado debido a la disminución de la inflación, algunos bancos centrales en países desarrollados han aumentado las tasas debido a la resistencia de la inflación en energéticos y servicios. En países emergentes, la inflación general disminuye, pero la subyacente baja lentamente debido a un mercado laboral fuerte. Las expectativas de inflación para Estados Unidos y México se ajustan, manteniéndose por encima de las metas de los bancos centrales.]

### Sección 3: Tipo de Cambio

La moneda mexicana se ha apreciado frente al dólar debido a factores como remesas, inversión extranjera, turismo y tasas de interés atractivas. Se espera que esta tendencia continúe y se pronostica un valor fundamental de cotización para el cierre del año.]

### Sección 4: Pronósticos Actinver

Se presentan los pronósticos para diferentes indicadores económicos en México y Estados Unidos, incluyendo el PIB, la inflación, la tasa de referencia y el tipo de cambio. Se observa un aumento en las expectativas de crecimiento en ambos países y un control gradual de la inflación en México.]

#### Análisis Integrado:

El informe destaca una perspectiva generalmente positiva para la economía tanto en México como en Estados Unidos. La demanda de manufacturas mexicanas por parte de empresas estadounidenses, combinada con un bajo desempleo y aumento de salarios, contribuye a un patrón sólido de consumo. Esto ha llevado a un aumento en las expectativas de crecimiento en ambos países.

Sin embargo, el análisis también subraya desafíos en términos de inflación. Mientras que en países desarrollados algunos bancos centrales han aumentado tasas debido a la resistencia de la inflación, en México la inflación está por encima de la meta del banco central. Esto podría tener implicaciones para las decisiones de política monetaria.]

#### Conclusiones y Recomendaciones:

En resumen, el análisis refleja oportunidades de crecimiento y desafíos en el panorama económico. Las recomendaciones se centran en gestionar riesgos, aprovechar oportunidades y estar preparados para cambios en las condiciones económicas.]

## ## Conclusiones y Recomendaciones

1. **Gestión de Inflación:** [Dada la persistencia de la inflación en México, las empresas deberían considerar estrategias para mitigar los riesgos inflacionarios en sus operaciones y planificación financiera.]
2. **Política Monetaria:** [Los inversores y tomadores de decisiones deben mantenerse atentos a las decisiones de política monetaria en ambos países, ya que podrían influir en las condiciones de financiamiento y tasas de interés.]
3. **Resiliencia Cambiaria:** [Dado el fortalecimiento de la moneda mexicana, las empresas que dependen de exportaciones deben monitorear el tipo de cambio y considerar estrategias para manejar posibles fluctuaciones]
4. **Vigilancia de Indicadores Clave:** [La demanda de manufacturas mexicanas ofrece oportunidades para el crecimiento económico. Las empresas deben buscar formas de aprovechar esta demanda y fortalecer sus cadenas de suministro]
5. **Flexibilidad y Previsión:** [Los mercados financieros son volátiles y pueden verse afectados por cambios repentinos. Las empresas y los inversionistas deben mantener flexibilidad y adaptabilidad en sus estrategias]

## Comentarios y conclusiones del caso de Estudio

En general, el proceso de obtener texto de un archivo PDF, usando OCR, es realmente muy eficiente, sin embargo, en esta práctica 2, tuve muchos problemas en poder guardar los resultados en formato PDF.

Mi idea inicial fue comparar dos contratos, uno en PDF original, y otro archivo PDF correspondiente al contrato firmado, a fin de identificar si ambos son idénticos o no. Conforme avancé, tuve demasiados errores de código que me obligaban a modificar el código proporcionado por CHATGPT. Al final, el resultado no fue el esperado por lo que decidí realizar este caso de estudio.

Mis conclusiones son por tanto, que conforme el nivel de sofisticación de una necesidad se incrementa, el código que proporciona CHATGPT no es cien por ciento funcional, por lo que debe complementarse la búsqueda con otras fuentes. Esto no implica que CHATGPT sea malo. De hecho, mi comentario es muy positivo sobre CHATGPT, pues en mi caso, tengo varios años de no ejercer la programación en forma cotidiana, y el poder lograr desarrollar un código para realizar una función en horas o inclusive un día, es sumamente mucho más eficiente, que si yo desarrollara el código por cuenta propia, haciendo búsquedas en internet. Seguramente me tomaría varios días más.

Finalmente, CHATGPT es un excelente complemento, guía y apoyo para

## 6. CONCLUSIÓN

CHATGPT resultó ser extremadamente eficiente en lo relacionado al análisis estadístico de datos. La información fue generada no solo en forma exacta, sino también, en forma muy rápida. Sin duda, puede considerarse a CHATGPT como una herramienta, más de apoyo, una herramienta que reemplaza a las personas que realizan este tipo de actividades. Evidentemente, financieros, desarrolladores de software, economistas u otros profesionistas que realizan estas tareas, deberán usar CHATGPT como herramienta de apoyo para poder generar análisis estadísticos mucho más complejos.

En lo relacionado al texto, CHATGPT funciona de forma también con un alto porcentaje de inteligencia para poder generar un análisis y resumen de información a un nivel licenciatura al menos.

Solo podría añadirse el hecho de que el código generado por CHATGPT generalmente funciona bien para tareas puntuales, pero en casos de proyectos más sofisticados, que impliquen un volumen grande archivos, o información, o pasos para completar tareas, será requerida la participación del programador para adaptar, modificar o corregir el código generado por CHATGPT. Sin duda esto no representa ningún impedimento. Al contrario, CHATGPT demuestra ser una de las mejores herramientas de apoyo.