

# Explorando las Capacidades de GitHub Copilot



Natali Lujan  
@natalilujanallende

# Natali Lujan Allende

Bachiller en Estadística Informática (UNALM) con especialización en  
Marketing Digital (UPC) y Maestría en Ciencia de Datos



**Women Techmakers Ambassador**



[.https://www.linkedin.com/in/natali-lujan-allende/](https://www.linkedin.com/in/natali-lujan-allende/)



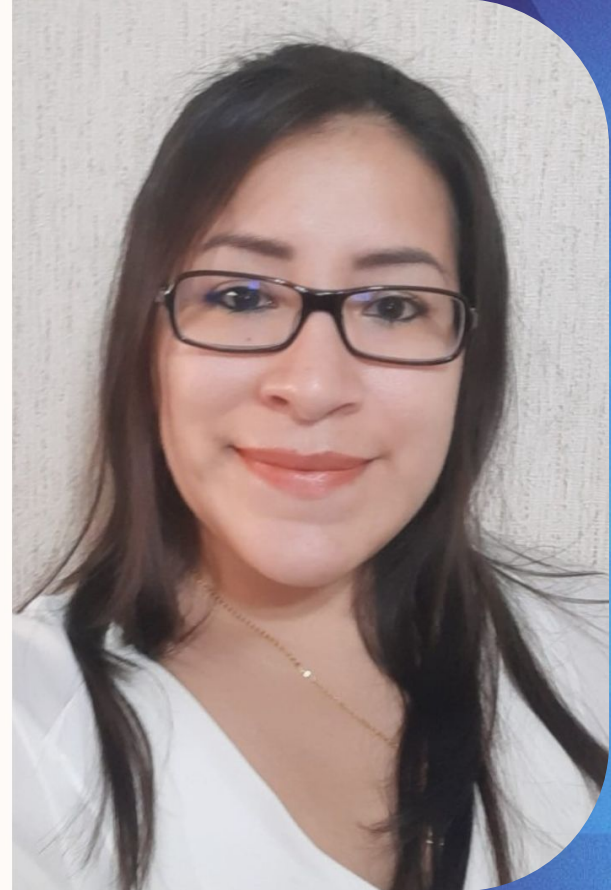
[.https://www.facebook.com/groups/biexpert](https://www.facebook.com/groups/biexpert)



[.https://www.linkedin.com/company/datacatnip/](https://www.linkedin.com/company/datacatnip/)



[.natalilujan@gmail.com](mailto:natalilujan@gmail.com)





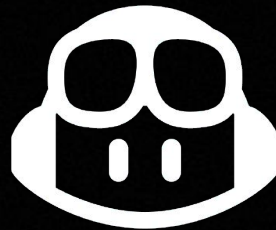
# Que es Github Copilot

Asistente de programación impulsado por inteligencia artificial desarrollado por GitHub y OpenAI.



## Características Clave

1. **Autocompletar Códigos:** Copilot puede predecir lo que el usuario quiere escribir y sugerir líneas de código o incluso funciones completas.
2. **Soporte Multilenguaje:** Es compatible con muchos lenguajes de programación, como JavaScript, Python, TypeScript, Ruby, Go, y más.
3. **Generación de Tests:** Puede ayudar a generar tests unitarios basados en el código que escribas, facilitando el proceso de pruebas.
4. **Documentación y Comentarios:** Copilot también entiende comentarios en lenguaje natural y sugiere código basado en ellos.





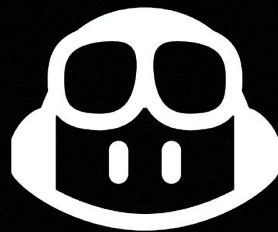
¿Por Qué Usar GitHub  
Copilot?





## ¿Por Qué Usar GitHub Copilot?

1. **Aumento de la Productividad:** Ayuda a reducir el tiempo de escritura de código al sugerir soluciones rápidas.
2. **Facilita el Aprendizaje:** Es útil para nuevos desarrolladores, ya que proporciona ejemplos de buenas prácticas.
3. **Colaboración:** Actúa como un compañero de codificación, lo que hace que el proceso de desarrollo sea más interactivo y menos tedioso.



**"¿Cómo puedo aprovechar al máximo GitHub Copilot para el análisis de datos?"**

# DEMO







## v1.195.0

★★★★☆ (1036)

[Switch to Pre-Release Version](#)

Extension is enabled on 'WSL: Ubuntu'

EXTENSION PACK

Free trial

# Your AI pair programmer

## Get Code Suggestions in real-time, right in your IDE

## Categories

## Programming Languages

# Machine Learning

## Education


## Snippets

## Resources



## Ask Copilot

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

 or type # to attach context

@ to chat with extensions

Type / to use commands

## GitHub Copilot

- **Uso Principal:** Su función principal es generar sugerencias de código automáticamente en función del contexto. Por ejemplo, si comienzas a escribir una función, Copilot intenta predecir y sugerir el resto del código.
- **Interacción:** No tiene una interfaz de chat. Las sugerencias aparecen directamente en el editor de código, y el desarrollador puede aceptar, rechazar o modificar estas sugerencias.

## GitHub Copilot Chat

- **Uso Principal:** GitHub Copilot Chat es ideal para realizar consultas complejas o resolver problemas específicos en tiempo real. Puedes hacer preguntas como “¿Cómo puedo optimizar este código?” o “¿Cuál es la mejor manera de manejar errores aquí?” y recibir una respuesta contextualizada.
- **Interacción:** Funciona como un asistente de conversación en el editor de código. Ofrece una experiencia similar a un chat, permitiéndote hacer preguntas en lenguaje natural y obtener respuestas o sugerencias detalladas.

## Diferencias entre GitHub Copilot y GitHub Copilot Chat

Característica	GitHub Copilot	GitHub Copilot Chat
Sugerencias de Código	Sí	Sí
Interfaz de Chat	No	Sí
Respuestas en Lenguaje Natural	No	Sí, permite hacer preguntas y recibir explicaciones
Interacción Directa	Aparece en el editor como sugerencias de código	A través de una ventana de chat
Mejor para	Autocompletado rápido y escritura de código	Resolver problemas y obtener explicaciones más detalladas

importar las bibliotecas necesarias para el análisis de  
datos



GitHub Copilot


analysis.py

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```





Press `Ctrl` + `I` to ask GitHub Copilot to do

 Generate



leer un archivo CSV en un DataFrame de pandas

Ask Copilot



Accept

Accept & Run



Discard







```
# Leer el archivo CSV en un DataFrame de pandas
df = pd.read_csv('ruta/al/archivo.csv')



# Mostrar las primeras filas del DataFrame
df.head()
```

Python

 Crear un nuevo dataframe con las columnas pageviews y newVisits agrupado por date

Ask Copilot

AcceptAccept & RunDiscard

```
df_grouped = df.groupby('date')[['pageviews', 'newVisits']].sum().reset_index()
print(df_grouped.head())
```

[ ]Python



crear un gráfico de dispersión de dos columnas (pageviews y newVisits ) usando matplotlib

Ask Copilot



Accept

Accept & Run

Discard



```
plt.scatter(df['pageviews'], df['newVisits'])
plt.xlabel('Pageviews')
plt.ylabel('New Visits')
plt.title('Scatter Plot of Pageviews vs New Visits')
plt.show()
```

Python



calcular estadísticas descriptivas del DataFrame

Ask Copilot



Accept

Accept & Run

Discard



```
df.describe()
```

crear y entrenar un modelo de regresión lineal con scikit-learn sabiendo que la variable respuesta es pageviews

Ask Copilot

Accept

Accept & Run

Discard



```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Definir las variables independientes (features) y la variable dependiente (target)
X = df_grouped[['newVisits', 'transactions', 'bounces', 'visits']]
y = df_grouped['pageviews']

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Crear el modelo de regresión lineal
model = LinearRegression()

# Entrenar el modelo
model.fit(X_train, y_train)

# Hacer predicciones en el conjunto de prueba
y_pred = model.predict(X_test)

# Evaluar el modelo
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')
```



```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Definir las variables independientes (features) y la variable dependiente (target)
X = df_grouped[['newVisits', 'transactions', 'bounces', 'visits']]
y = df_grouped['pageviews']

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Crear el modelo de regresión lineal
model = LinearRegression()

# Entrenar el modelo
model.fit(X_train, y_train)

# Hacer predicciones en el conjunto de prueba
y_pred = model.predict(X_test)

# Evaluar el modelo
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')
```

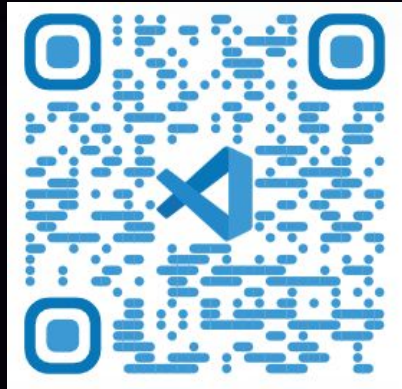
[13] ✓ 1.4s

Python

... Mean Squared Error: 326573.4998899098  
R^2 Score: 0.8755558714539844

# Gracias!

[aka.ms/learn-copilot](https://aka.ms/learn-copilot)



Natali Lujan  
@nataliperu