Análisis de Tendencias de Lenguajes de Programación y su Influencia en la Educación y el Mercado Laboral

Integrantes

Rafael Acosta Márquez , 311

Eisler Francisco Valles Rodríguez , 311

Introducción

El presente informe detalla el análisis de la evolución y proyección de popularidad de varios lenguajes de programación basándose en sus contribuciones en GitHub, correlacionando estos datos con las tendencias de búsqueda en Google Trends y StackOverflow. Para realizar este análisis, se ha empleado una metodología estadística basada en la regresión lineal, con el objetivo de modelar y predecir el comportamiento futuro de estas contribuciones a lo largo del tiempo. Además se realizaron pruebas de hipótesis para observar la influencia de la popularidad de estos lenguajes en la aparición en los currículums laborales.

Regresión lineal

Metodología Estadística

La regresión lineal es un enfoque estadístico que permite entender y modelar la relación entre dos variables continuas. En este caso, se ha aplicado una regresión lineal para cada lenguaje de programación, utilizando como variable independiente el tiempo (años) y como variable dependiente el número de contribuciones en GitHub, búsquedas en Google Trends y StackOverflow.

Los pasos seguidos en la metodología fueron:

- Recolección de Datos: Se extrajeron los datos históricos de contribuciones en GitHub, las búsquedas relacionadas en Google Trends y StackOverflow de los últimos 5 años para los lenguajes C#, C++, Python, Go y JavaScript.

- Preparación de Datos: Los datos fueron limpiados y formateados adecuadamente para garantizar su calidad y precisión antes de ser utilizados en el análisis.

- Modelado Estadístico: Se implementaron modelos de regresión lineal por separado para cada lenguaje de programación, asumiendo una relación lineal entre el año y las contribuciones en GitHub, búsquedas en GoogleTrends y StackOverflow.

- Verificación de Supuestos: Se comprobó que los supuestos subyacentes a la regresión lineal fueran válidos para cada conjunto de datos, incluyendo la independencia de los residuos, homocedasticidad, y normalidad.

- Estimación de Parámetros: Se calcularon los parámetros de la regresión lineal, es decir, la pendiente y la intersección, que describen la tendencia en las contribuciones.

- Análisis de Resultados: Se interpretaron los coeficientes estimados para evaluar el grado y dirección de la relación entre el tiempo y las contribuciones.

- Proyección Futura: Utilizando los modelos de regresión lineal, se extrapolaron las tendencias para hacer proyecciones de las contribuciones para los años 2024 y 2025.

Nota

Durante la fase de verificación de supuestos, nuestros esfuerzos se orientaron a asegurar que las condiciones fundamentales para la regresión lineal se mantuvieran intactas. Hemos evaluado la independencia de los residuos, la homocedasticidad y la normalidad para cada uno de los conjuntos de datos asociados con los lenguajes de programación. Sin embargo, debemos reconocer que no todos los supuestos se han cumplido de manera estricta en nuestros análisis preliminares.

A pesar de la presencia de ciertas desviaciones en los supuestos teóricos, hemos decidido utilizar los datos actuales como base para este proyecto, sirviendo como una muestra representativa de cómo se debe realizar un análisis más amplio. La información obtenida de los modelos actuales, aunque limitada por el volumen de datos, proporciona una visión valiosa de las tendencias de los lenguajes de programación en el contexto de popularidad.

Por tanto, aunque los modelos actuales no cumplen plenamente con los supuestos de la regresión lineal, nos proporcionan un punto de partida sólido para una investigación más exhaustiva en la que se puedan abordar estas limitaciones y mejorar la precisión de nuestros modelos predictivos.

Datos Utilizados

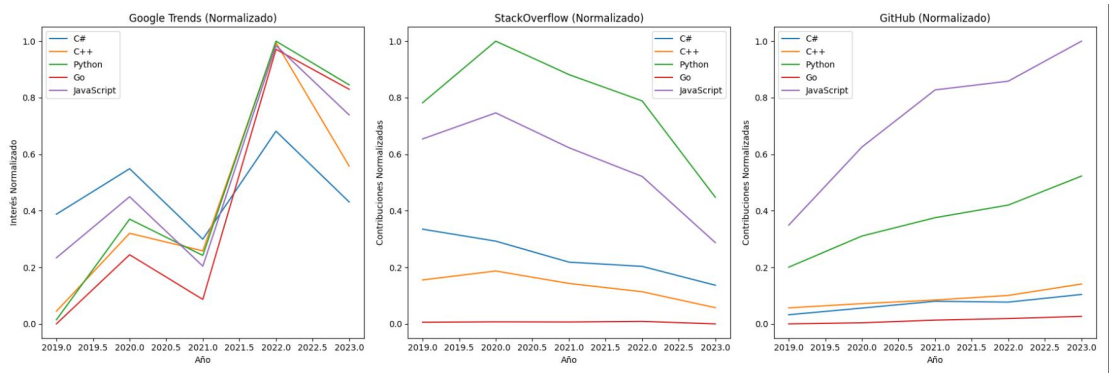
Descripción de los Data Sets

Google Trends
- Origen de los datos: Los datos de Google Trends representan el volumen de búsquedas para los lenguajes de programación específicos. Estos valores pueden interpretarse como un indicativo del interés en cada lenguaje durante un tiempo determinado.
- Periodo de tiempo que cubren: 2019 a 2023.
- Variables específicas recopiladas: Volumen de búsquedas para los lenguajes de programación C#, C++, Python, Go, y JavaScript.

StackOverflow
- Origen de los datos: Los datos de StackOverflow representan el número de preguntas o discusiones relacionadas con cada lenguaje de programación, reflejando la actividad comunitaria y el interés en solucionar problemas específicos.
- Periodo de tiempo que cubren: 2019 a 2023.
- Variables específicas recopiladas: Número de contribuciones o preguntas para los lenguajes de programación C#, C++, Python, Go, y JavaScript.

GitHub
- Origen de los datos: Los datos de GitHub reflejan el número de contribuciones (como commits, pull requests, etc.) asociados a repositorios que utilizan los lenguajes de programación específicos. Estos indican la actividad de desarrollo y el interés en construir proyectos con estos lenguajes.
- Periodo de tiempo que cubren: 2019 a 2023.
- Variables específicas recopiladas: Número de contribuciones en GitHub para los lenguajes de programación C#, C++, Python, Go, y JavaScript.

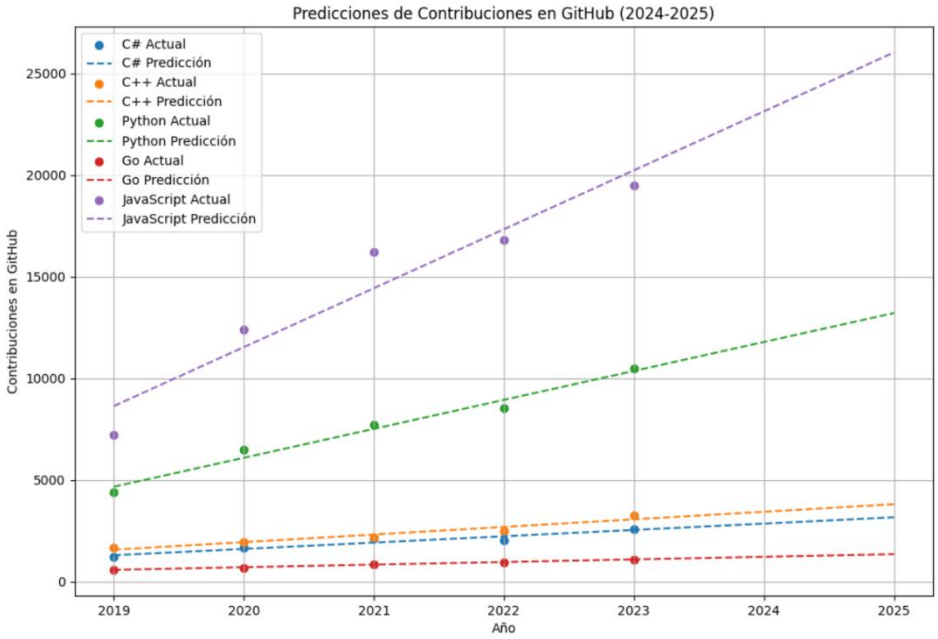
Gráfico de los datos normalizados



Aplicación de la Regresión Lineal para observar tendencias de estos lenguajes en los años 2024 y 2025

Los siguientes gráficos muestran las tendencias y predicciones de las contribuciones en distintas plataformas (GitHub, StackOverflow y Google Trends) para varios lenguajes de programación (C#, C++, Python, Go y JavaScript) a lo largo de un periodo que incluye los años 2019 a 2025.

Regresión Lineal Github



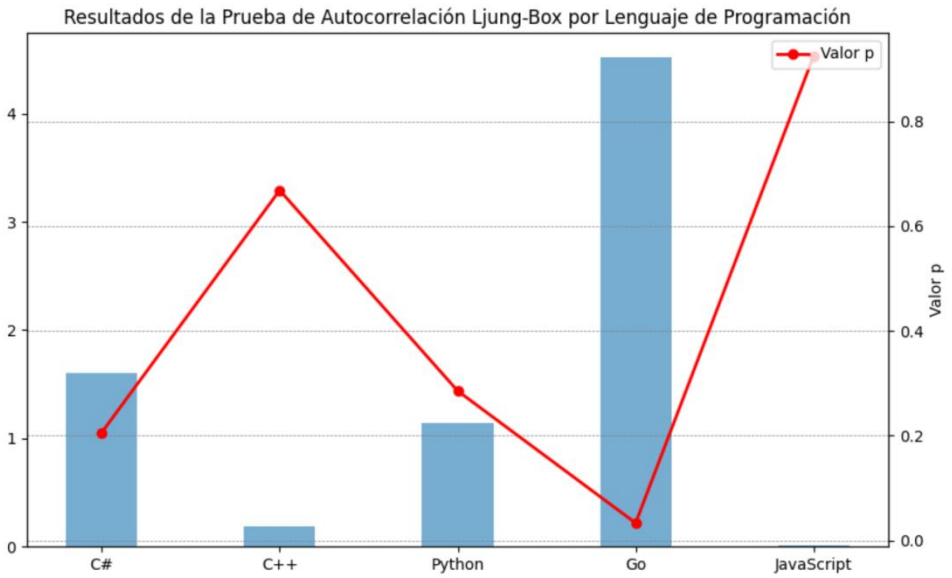
Predicciones de Contribuciones en GitHub (2024-2025)

- C#: Se observa un crecimiento constante en las contribuciones reales y se espera que la tendencia siga aumentando según la predicción.
- C++: Las contribuciones muestran una tendencia constante a lo largo del tiempo, sin grandes incrementos ni disminuciones.
- Python: Exhibe un crecimiento exponencial tanto en las contribuciones reales como en las predicciones, lo que podría reflejar la creciente popularidad y uso de Python en el desarrollo.
- Go: Las contribuciones reales son menores en comparación con otros lenguajes, pero la predicción muestra un crecimiento constante.
- JavaScript: Las contribuciones reales son altas y se prevé que la tendencia creciente continúe en el futuro.

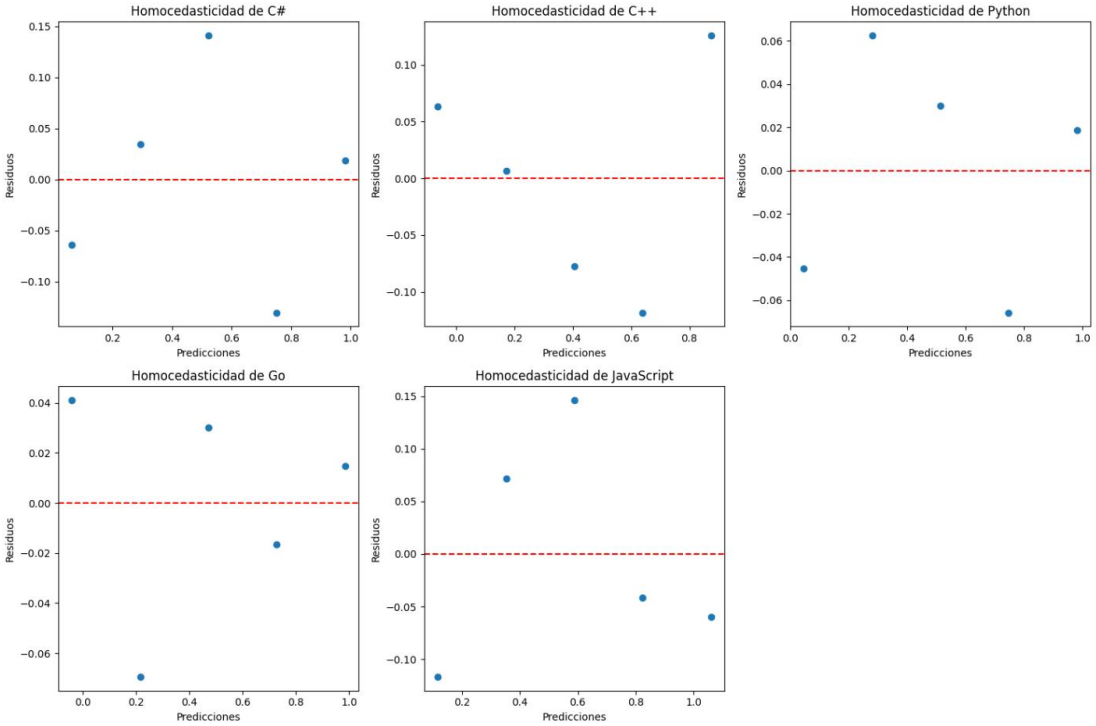
Fórmulas correspondientes
- C#: y = 311.50x + -627621.90
- C++: y = 373.40x + -752324.60
- Python: y= 1425.20x + -2872811.40
- Go: y = 128.30x + -258458.30
- JavaScript: y = 2900.80x + -5848077.00

Verificación de Supuestos

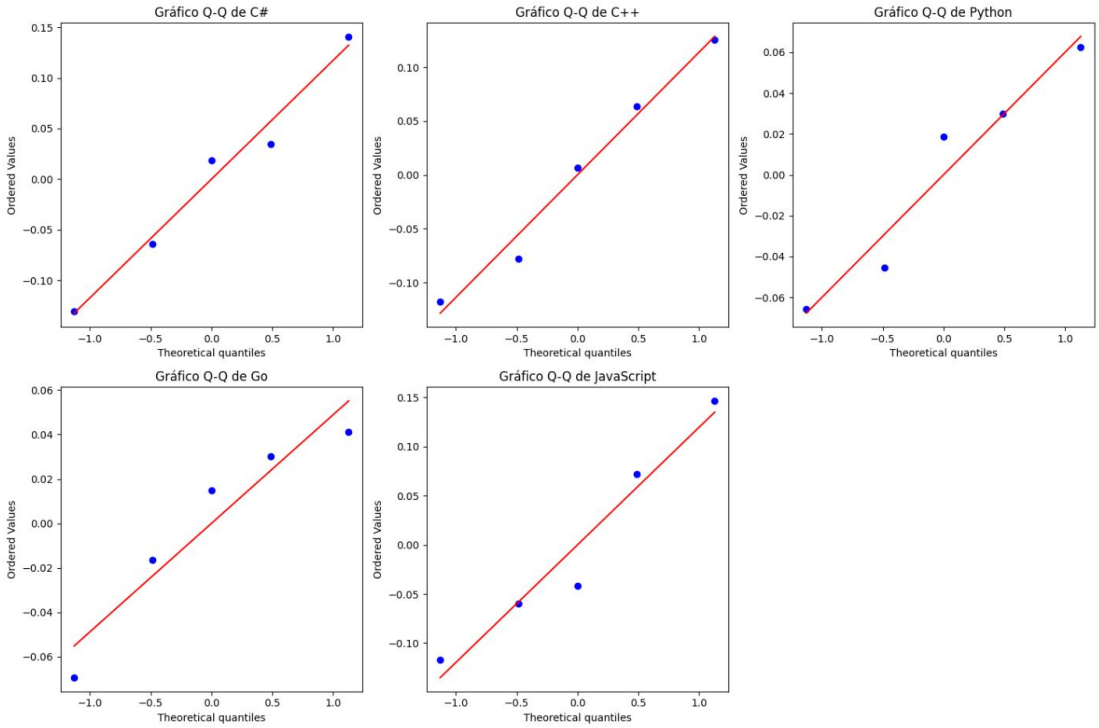
- Los errores (e1, … , en) son independientes.



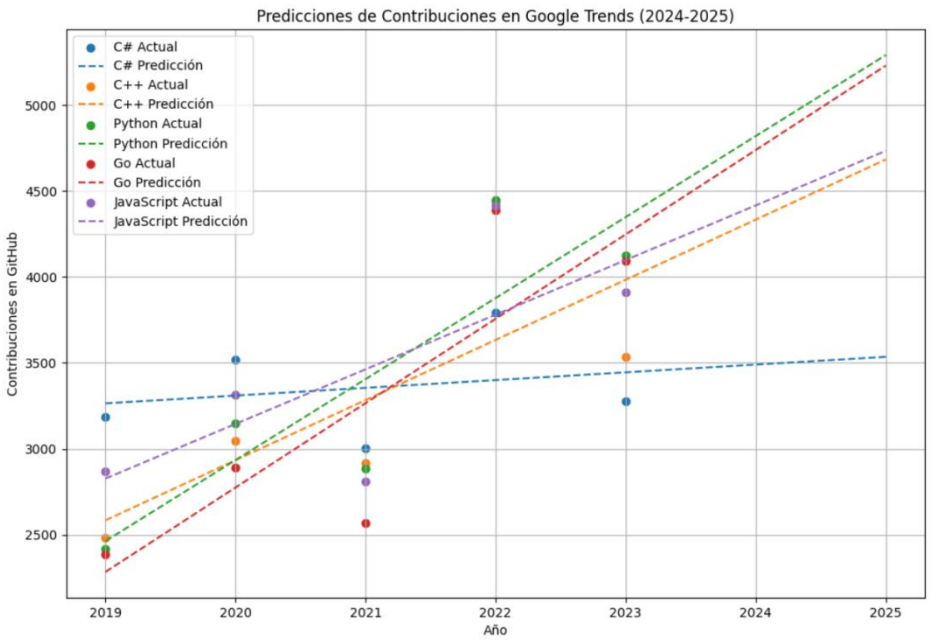
- El valor esperado del error aleatorio ei es cero (E(ei) = 0).
- La varianza del error aleatorio es constante (Homocedasticidad).



- Los errores son idénticamente distribuidos y siguen una distribución normal (Normalidad).



Regresión Lineal GoogleTrends



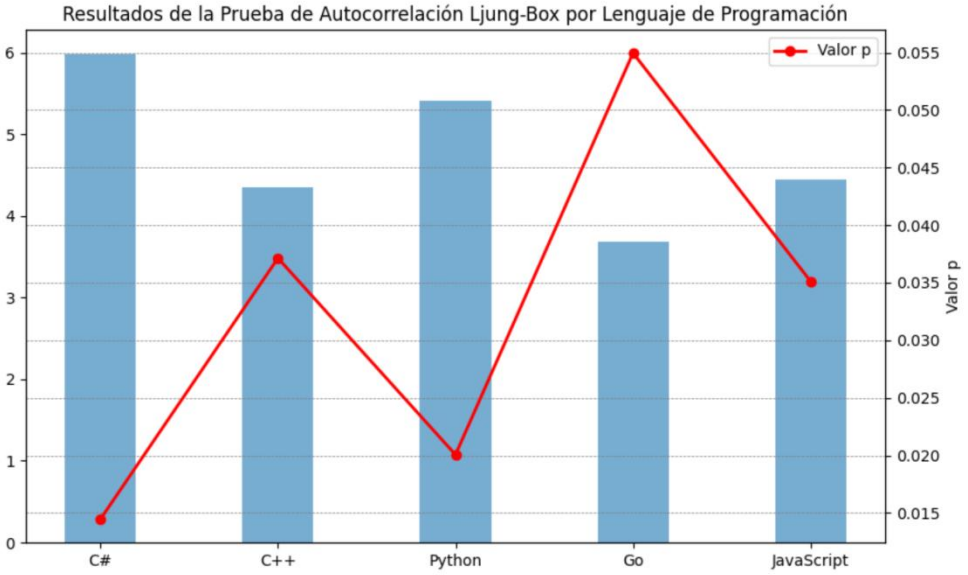
Predicciones de Contribuciones en Google Trends (2024-2025)

- C#: Las contribuciones reales y las predicciones indican un crecimiento lineal moderado.
- C++: Aunque las contribuciones reales son relativamente bajas, las predicciones sugieren un crecimiento constante.
- Python: Muestra una tendencia ascendente fuerte, tanto en las contribuciones reales como en las predicciones, destacando su creciente interés y popularidad.
- Go: A pesar de un comienzo bajo en comparación con otros lenguajes, Go muestra un aumento sostenido en el interés, según Google Trends.
- JavaScript: Se ve un crecimiento sostenido en las contribuciones reales, con expectativas de que la tendencia se mantenga en aumento.

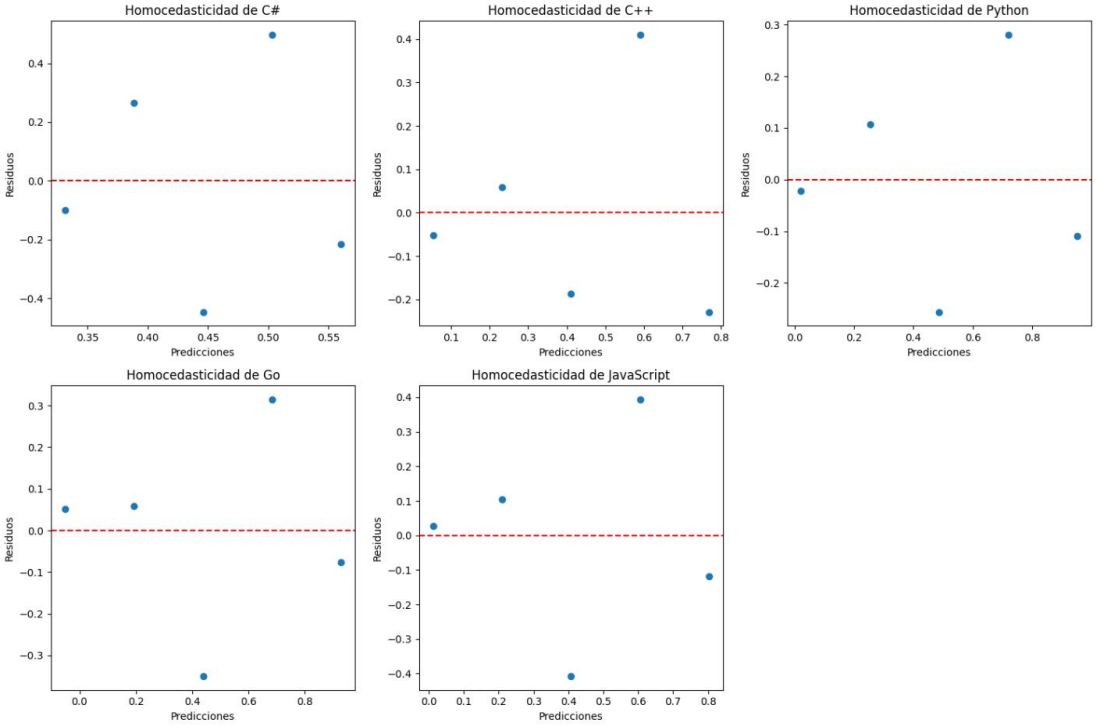
Fórmulas correspondientes
- C#: y = 45.10x + -87792.70
- C++: y = 349.90x + -703864.70
- Python: y = 471.30x + -949092.10
- Go: y = 490.90x + -988843.70
- JavaScript: y = 317.80x + -638811.60

Verificación de Supuestos

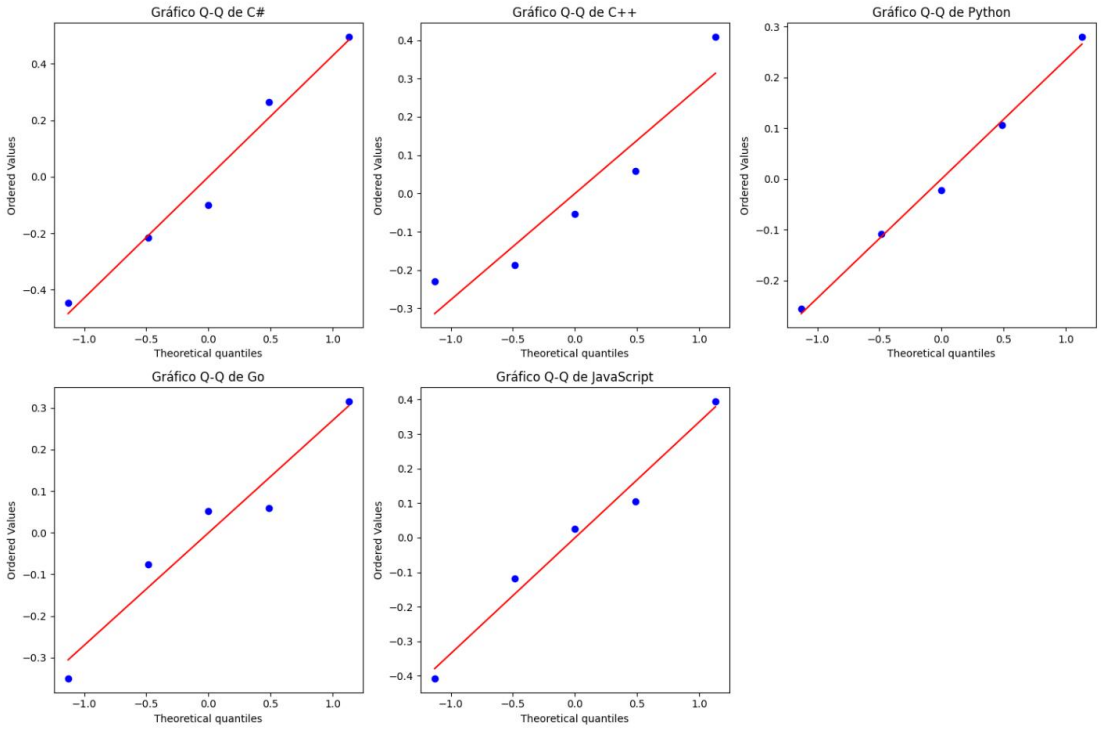
- Los errores (e1, … , en) son independientes.



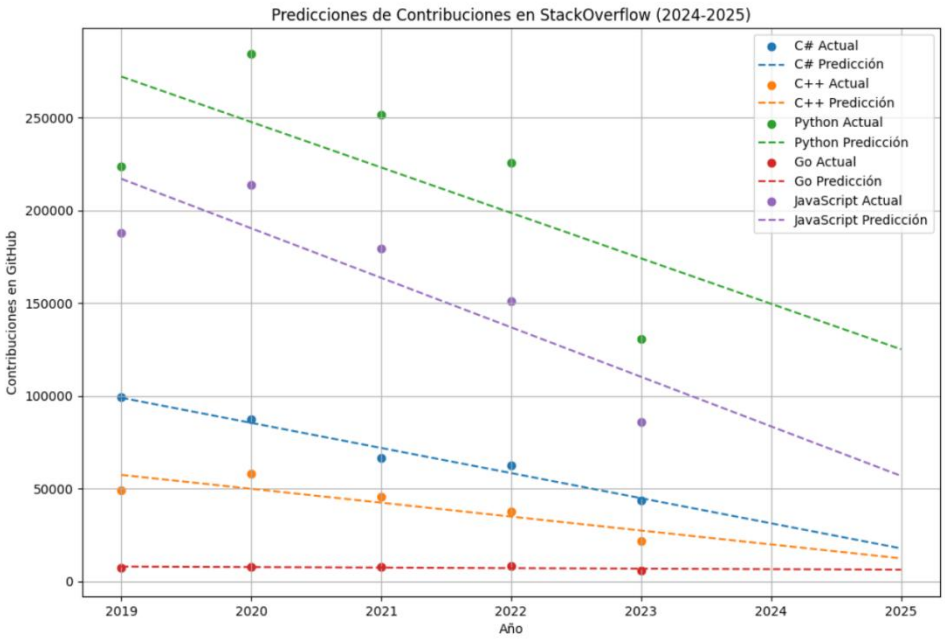
- El valor esperado del error aleatorio ei es cero (E(ei) = 0).
- La varianza del error aleatorio es constante (Homocedasticidad).



- Los errores son idénticamente distribuidos y siguen una distribución normal (Normalidad).



Regresión Lineal StackOverflow



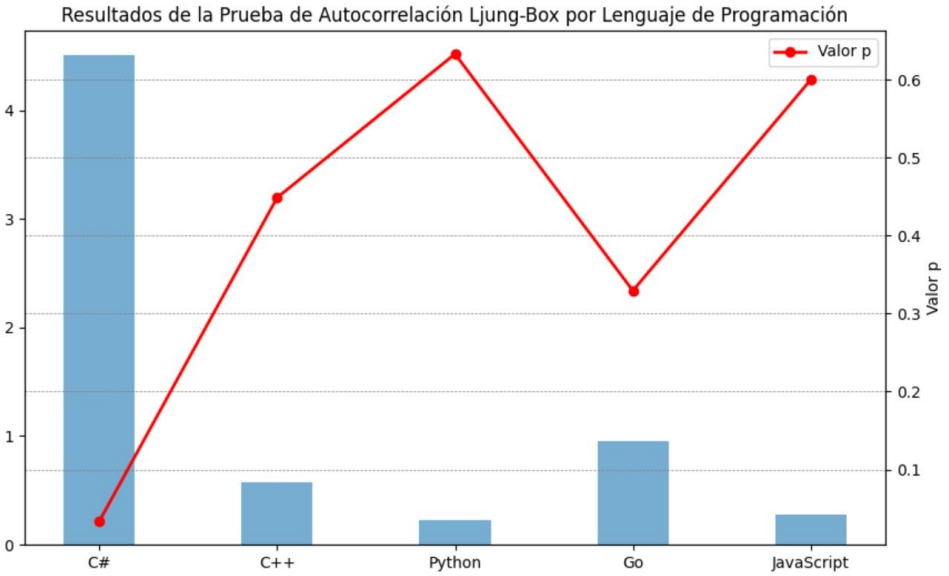
Predicciones de Contribuciones en StackOverflow (2024-2025)

- C#: Muestra una disminución progresiva tanto en las contribuciones reales como en las predicciones.
- C++: Las contribuciones reales y las predicciones muestran un descenso, lo que puede indicar una disminución en la popularidad o en el uso de C++ en StackOverflow.
- Python: Al igual que en GitHub, Python muestra un aumento significativo en las contribuciones reales y en las predicciones, lo que refleja su sólido crecimiento en la comunidad.
- Go: Presenta una disminución en las contribuciones en StackOverflow, tanto en datos reales como en las predicciones.
- JavaScript: Se observa un decrecimiento en las contribuciones reales, y se espera que esta tendencia a la baja continúe.

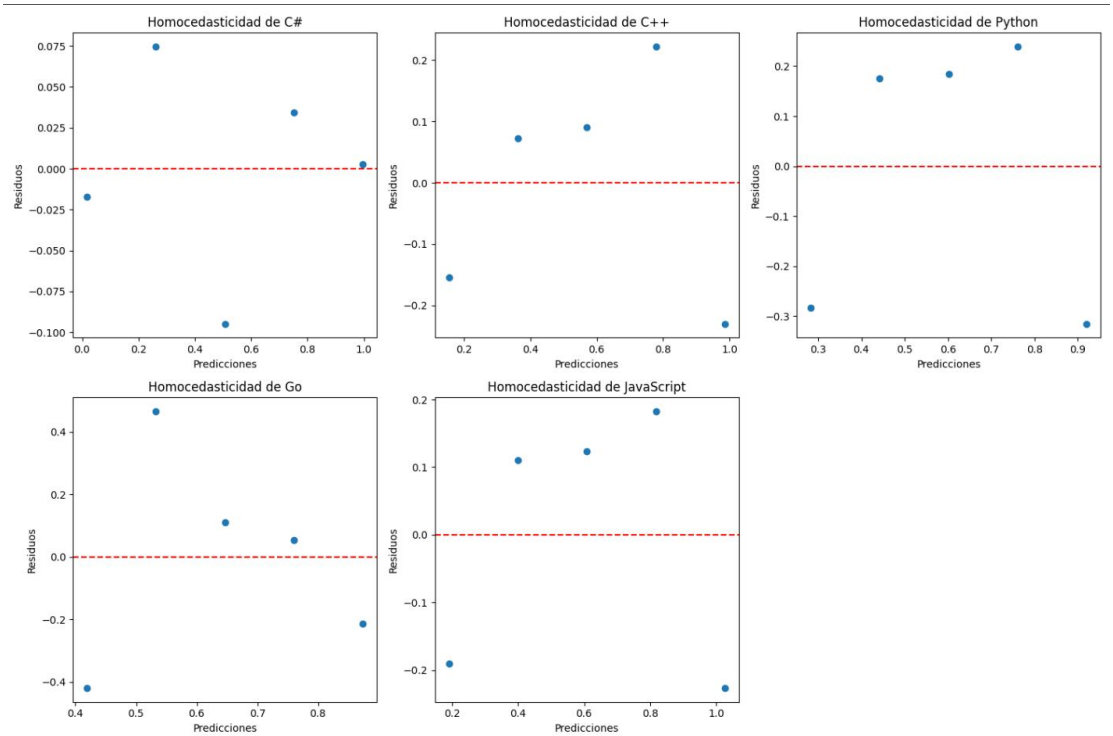
Fórmulas correspondientes
- C#: y = -13532.20x + 27420569.40
- C++: y = -7496.10x + 15192121.90
- Python: y = -24496.70x + 49730990.70
- Go: y = -278.10x + 569568.30
- JavaScript: y = -26686.10x + 54096293.50

Verificación de Supuestos

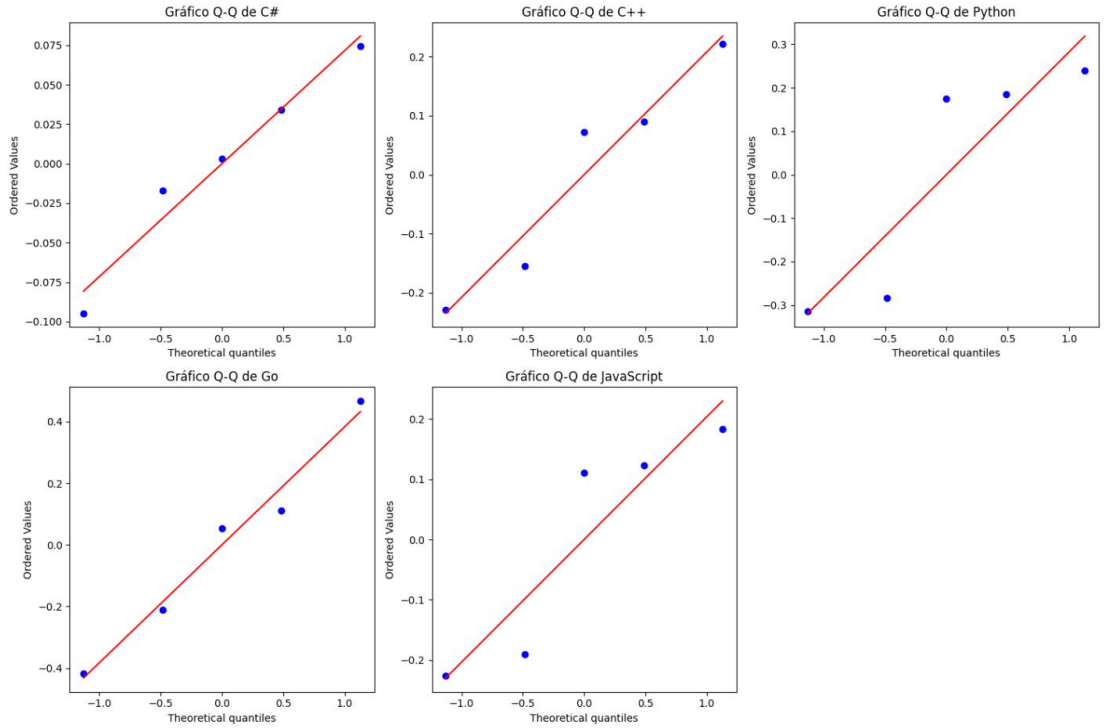
- Los errores (e1, … , en) son independientes.



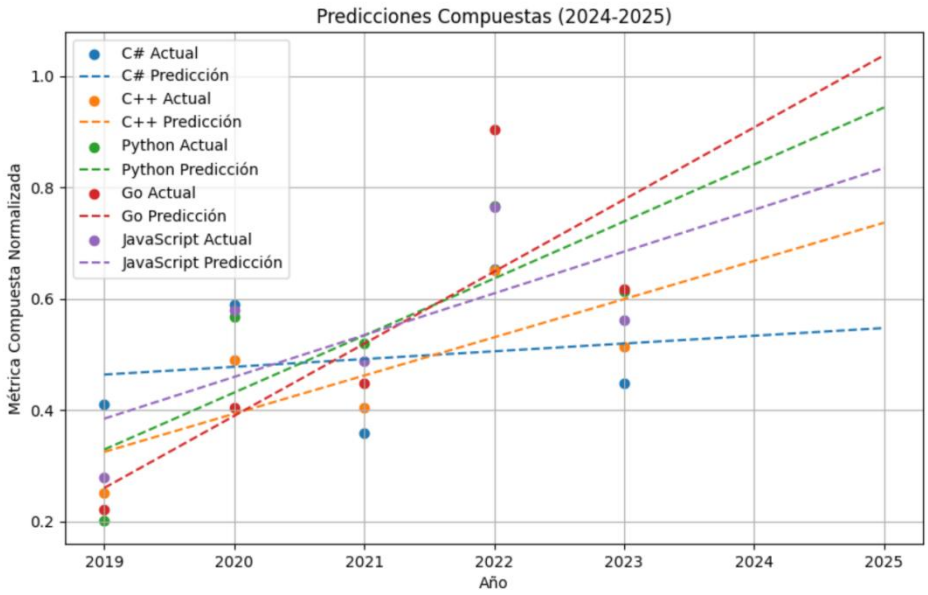
- El valor esperado del error aleatorio ei es cero (E(ei) = 0).
- La varianza del error aleatorio es constante (Homocedasticidad)



- Los errores son idénticamente distribuidos y siguen una distribución normal (Normalidad).



Regresión Lineal Compuesta



Predicciones Compuestas (2024-2025):
Este gráfico muestra una métrica compuesta tomando en cuenta las tres fuentes de datos (Google Trends, Stack Overflow y GitHub). Las tendencias indican que Python y Go podrían tener un crecimiento sostenido y más rápido que los otros lenguajes hacia 2025. JavaScript, C++ y C# también muestran crecimiento, pero a un ritmo más lento.

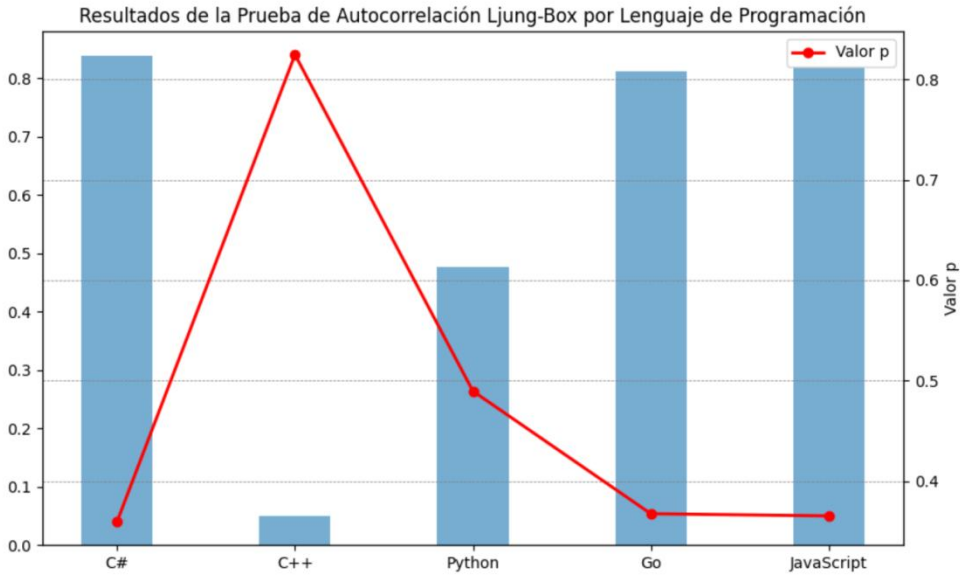
Interpretación general:

- Python parece ser el líder claro en todas las métricas hacia 2025, lo que indica su fuerte presencia y crecimiento continuo en el desarrollo de software, la comunidad y el interés de búsqueda.
- JavaScript también muestra un desempeño sólido, particularmente en GitHub y Google Trends, lo cual es esperable dado su papel central en el desarrollo web.
- C# y C++ muestran un crecimiento más moderado pero constante, lo que sugiere una adopción sostenida y una base de usuarios estable.
- Go tiene un crecimiento notable en la métrica compuesta, aunque no es el líder en ninguna de las plataformas individuales, lo que puede indicar una tendencia emergente o una adopción creciente en nichos específicos de la industria.

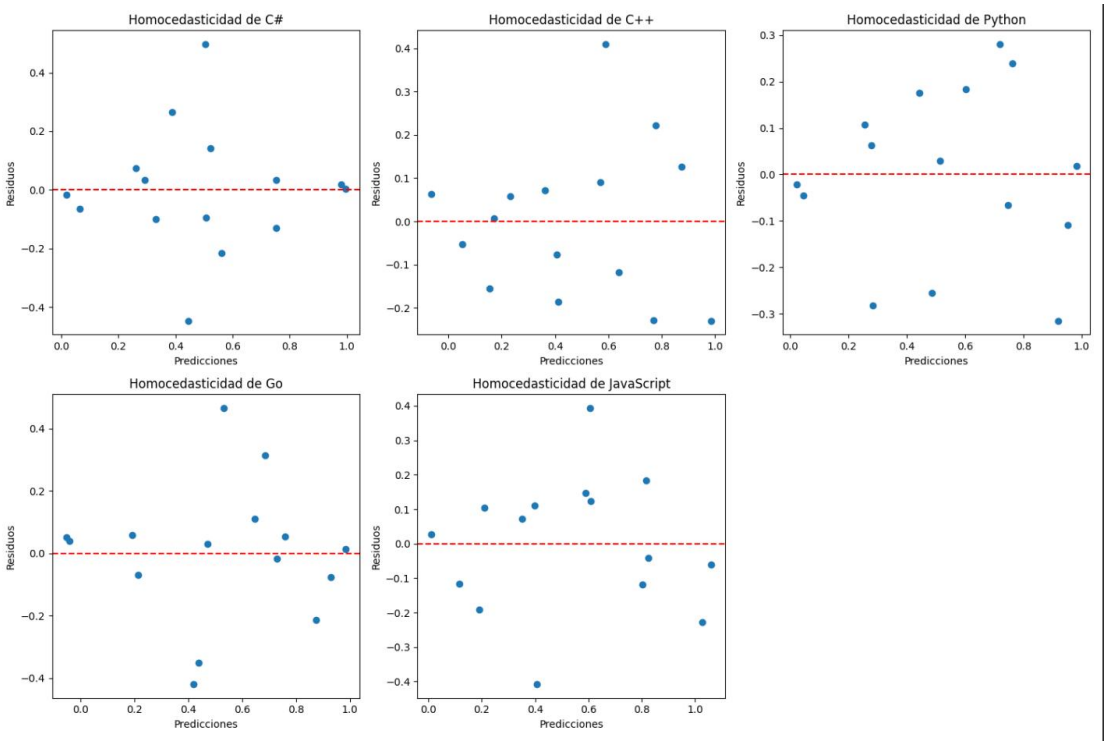
Fórmulas correspondientes
- C#: y = 0.01x + -27.66
- C++: y = 0.07x + -138.18
- Python: y = 0.10x + -206.41
- Go: y = 0.13x + -261.20
- JavaScript: y = 0.08x + -151.06

Verificación de Supuestos

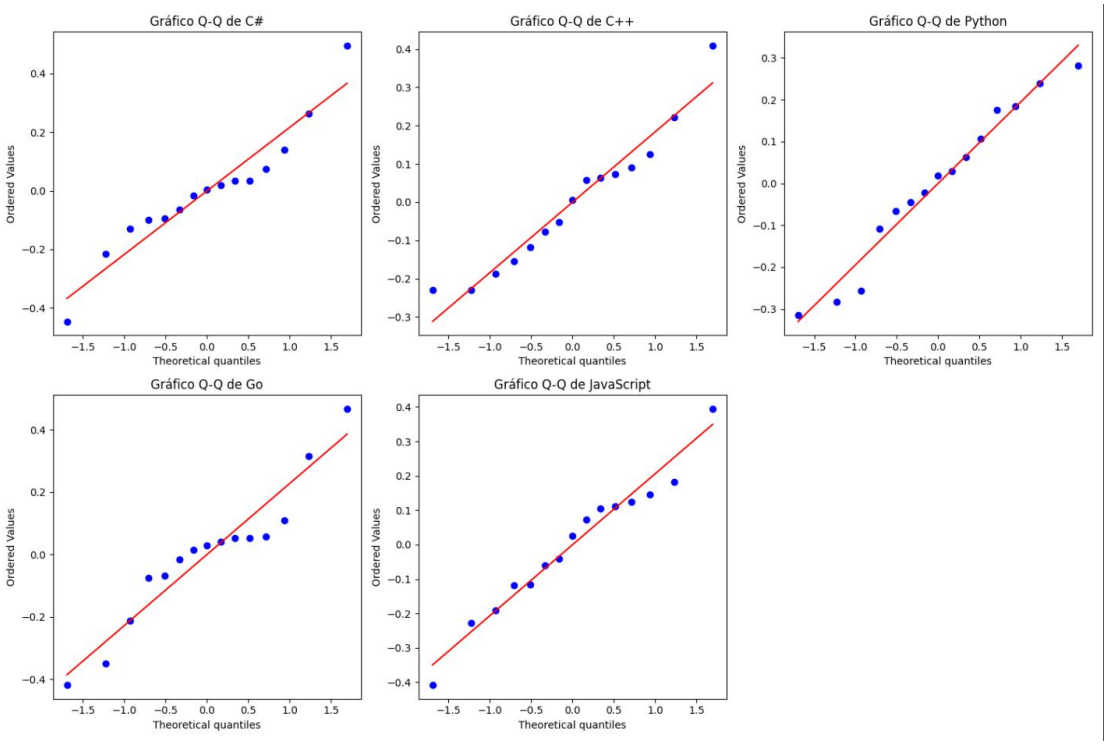
- Los errores (e1, … , en) son independientes.



- El valor esperado del error aleatorio ei es cero (E(ei) = 0).
- La varianza del error aleatorio es constante (Homocedasticidad).



- Los errores son idénticamente distribuidos y siguen una distribución normal (Normalidad).



El caso de Go en los gráficos muestra un fenómeno interesante. Aunque no lidera en las métricas individuales de Google Trends, Stack Overflow o GitHub, su rendimiento combinado en la métrica compuesta muestra un crecimiento más rápido que otros lenguajes como C# y C++.

Posibles explicaciones para este patrón:

- Crecimiento Balanceado: Go podría no ser el líder en ninguna plataforma individualmente, pero si muestra un crecimiento consistente y balanceado en todas ellas, esto podría resultar en una métrica compuesta alta. Es decir, su fortaleza puede estar en su desempeño general en lugar de su dominio en una sola área.
- Adopción en Nichos Específicos: Go es conocido por su eficiencia y rendimiento en sistemas de cómputo en la nube y desarrollo de servicios backend. El crecimiento en la métrica compuesta podría estar reflejando una adopción creciente en estos nichos donde Go es particularmente favorecido.
- Maduración de la Tecnología: A medida que un lenguaje madura, sus mejoras pueden comenzar a alinearse más estrechamente con las necesidades de la industria, lo que podría estar ocurriendo con Go. Su diseño enfocado en la concurrencia y el rendimiento puede estar atrayendo más proyectos y desarrolladores conforme estas características se vuelven más relevantes.
- Cambio en Tendencias de Desarrollo: Los patrones de adopción de tecnología pueden cambiar debido a la aparición de nuevas prácticas y paradigmas. Por ejemplo, la creciente popularidad de la arquitectura de microservicios podría favorecer el uso de Go, impulsando su crecimiento en todas las métricas.
- Comunidad Activa y Recursos de Aprendizaje: Una comunidad de desarrolladores activa y la disponibilidad de recursos de aprendizaje pueden haber ayudado a Go a crecer de manera más uniforme en comparación con otros lenguajes que quizás tienen picos y valles más pronunciados en su popularidad o uso.

Explicación de las fórmulas

En las fórmulas antes expuestas y representa las contribuciones estimadas en Github, GoogleTrends, StackOverflow, GoogleTrends y todas juntas respectivamente para el año x. La variable x es el año de interés, la pendiente (el coeficiente antes de x) indica el cambio anual en las contribuciones, y el término de intersección (el número sin x al lado) representa el punto de partida teórico en el año 0 (es decir, el punto en el que la línea de regresión cruza el eje y)

Interpretación General de los Supuestos

- Los errores (e1, … , en) son independientes.

Los gráficos muestran los resultados de la Prueba de Autocorrelación Ljung-Box aplicada a diferentes lenguajes de programación, con el eje vertical representando el valor p de la prueba y el eje horizontal mostrando los lenguajes de programación evaluados.

La Prueba de Ljung-Box es una herramienta estadística que se utiliza para comprobar si hay autocorrelación en un conjunto de datos en diferentes retrasos.

Este supuesto, se relaciona con la hipótesis nula de esta prueba. Si los errores son independientes, esperaríamos que el valor p de la prueba sea alto (típicamente mayor que un umbral como 0.05 o 0.01), lo que indicaría que no hay suficiente evidencia para rechazar la hipótesis nula y que los errores pueden ser considerados independientes.

Interpretación general de estos gráficos:

- C#: Muestra un valor p muy bajo, cerca de cero, lo cual indica que hay evidencia de autocorrelación entre los errores y que la hipótesis nula de independencia de errores puede ser rechazada para este lenguaje de programación.
- C++: Presenta el valor p más alto, superando el umbral de 0.8, lo que sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de independencia de errores.
- Python: Tiene un valor p intermedio, alrededor de 0.4, que aún es relativamente alto y puede indicar que los errores son independientes.
- Go: Al igual que C#, muestra un valor p cerca de cero, lo que indica una autocorrelación significativa y la posibilidad de rechazar la hipótesis nula de independencia de errores.
- JavaScript: Presenta un valor p que está en el límite, ligeramente por debajo de 0.5, la decisión de rechazar o no la hipótesis nula es incierta

Los datos sugieren que solo en C++ no hay suficiente evidencia para rechazar la hipótesis nula de independencia de errores. En los casos de C# y Go, parece haber una fuerte evidencia de autocorrelación, lo que significaría que los errores no son independientes. Python y JavaScript están en una zona intermedia y la conclusión dependerá del nivel de significancia que se aplicó (Bajo).

- El valor esperado del error aleatorio ei es cero (E(ei) = 0).

Este supuesto se verifica durante la estimación del modelo de regresión lineal. Los modelos de regresión lineal implementados en el paquete estadístico utilizado(scikit-learn), están diseñados para asegurar que la media de los errores sea cero. La verificación manual seria calcular la media de los residuos y comprobar que sea cercana a cero.

- La varianza del error aleatorio es constante (Homocedasticidad).

Los gráficos muestran una serie de diagramas de dispersión que se utilizan para evaluar la homocedasticidad, es decir, la constancia de la varianza de los errores en diferentes modelos de regresión, cada uno correspondiente a un lenguaje de programación específico: C#, C++, Python, Go y JavaScript.

La homocedasticidad es una suposición importante en la regresión lineal, que indica que la varianza de los términos de error es la misma en todos los niveles de la variable independiente.

Para cada subgráfico:
- El eje horizontal (Predicciones) representa los valores predichos por el modelo de regresión.
- El eje vertical (Residuos) representa los residuos, que son las diferencias entre los valores observados y los valores predichos por el modelo.

Si la varianza de los errores es constante (cumpliendo con la suposición de homocedasticidad), los puntos deben estar distribuidos aleatoriamente alrededor del eje horizontal (cero en el eje de residuos) sin formar patrones discernibles y sin mostrar embudo o formas expansivas o contractivas.

Interpretación general de estos gráficos:

- C#: Los residuos parecen estar dispersos aleatoriamente alrededor de la línea horizontal, sin un patrón claro de expansión o contracción, lo cual es indicativo de homocedasticidad.
- C++: Los puntos muestran una dispersión variable y algunos parecen alejarse de la línea horizontal a medida que aumentan las predicciones, lo que podría sugerir la presencia de heterocedasticidad
- Python: Similar a C++, hay una dispersión que podría sugerir un patrón de heterocedasticidad, con los residuos mostrando una tendencia a dispersarse más para valores predichos más altos.
- Go: Hay algunos puntos que están más alejados de la línea horizontal, pero en general, no parece haber un patrón claro que sugiera heterocedasticidad.
- JavaScript: No hay un patrón claro de expansión o contracción, pero la variación parece ser ligeramente mayor en los valores predichos más altos.

Los gráficos sugieren que los modelos para C# y Go podrían cumplir con la suposición de homocedasticidad, mientras que los modelos para C++, Python y JavaScript podrían no cumplirla.

- Los errores son idénticamente distribuidos y siguen una distribución normal (Normalidad).

Los gráficos son un conjunto de gráficos Q-Q (Cuantil-Cuantil), que se utilizan para evaluar si un conjunto de datos sigue una distribución específica, que en este caso es la distribución normal. Cada gráfico Q-Q compara los cuantiles teóricos de una distribución normal con los cuantiles de los errores de un modelo de regresión para cada lenguaje de programación: C#, C++, Python, Go y JavaScript.

En un gráfico Q-Q, si los puntos (que representan los cuantiles de los datos) siguen la línea roja (que representa los cuantiles de una distribución normal), entonces los datos se consideran aproximadamente normalmente distribuidos.

Interpretación general de estos gráficos:

- C#: Los puntos siguen la línea bastante de cerca, lo que sugiere que los errores están aproximadamente normalmente distribuidos.
- C++: Los puntos también siguen la línea roja estrechamente, excepto quizás en los extremos, lo que indica que los errores son aproximadamente normales con algunas desviaciones en los valores extremos.
- Python: Los puntos siguen la línea muy de cerca, lo que indica que los errores están muy cerca de una distribución normal.
- Go: Los puntos siguen la línea con alguna variación, pero no hay desviaciones claras que sugieran una desviación significativa de la normalidad.
- JavaScript: Similar a Go, los puntos siguen la línea roja con algunas variaciones pequeñas, sugiriendo que los errores son aproximadamente normales.

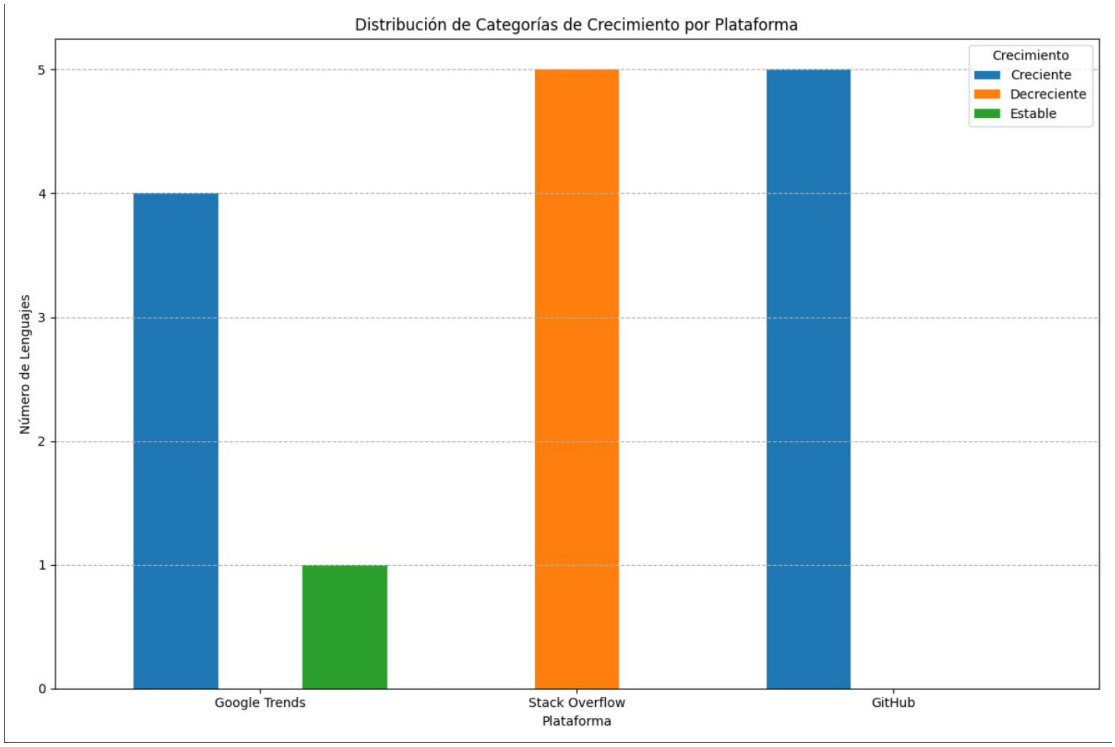
Los errores de los modelos para cada lenguaje de programación son aproximadamente normalmente distribuidos, cumpliendo con el supuesto de normalidad. Las pequeñas desviaciones en los extremos (como se ve en algunos gráficos) son comunes, especialmente en conjuntos de datos más pequeños o con valores atípicos. Sin embargo, no hay desviaciones sistemáticas grandes que sugieran una violación grave de la normalidad.

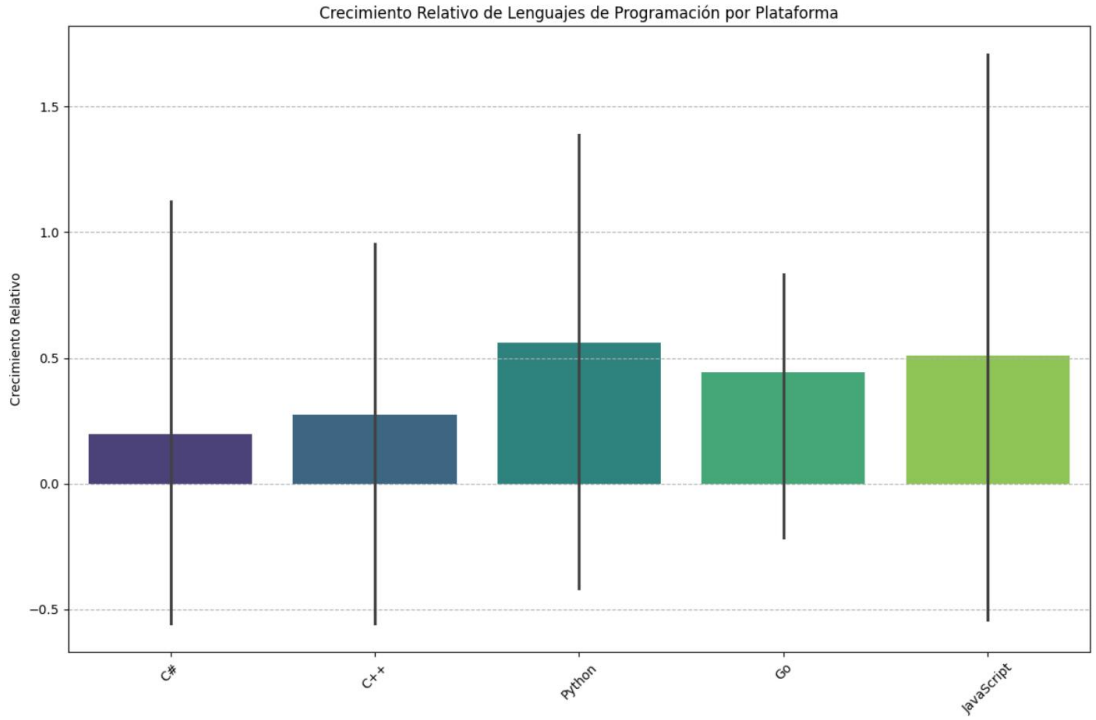
Influencia en la Educación en Ciencias de la Computación

Hipótesis:
H0: No hay diferencia en la distribución de las categorías de crecimiento entre las plataformas.
H1: Hay diferencia en la distribución de las categorías de crecimiento entre las plataformas.

Analizar la inclusión de estos lenguajes en programas académicos a lo largo del tiempo.
Para investigar cómo las tendencias afectan los currículos académicos usando una prueba de chi-cuadrado, primero necesitamos definir las categorías y cómo se relacionan estas tendencias con la inclusión en los currículos. Dado que no tenemos datos directos sobre currículos académicos, vamos a simplificar el escenario y asumir que el interés en enseñar un lenguaje de programación en el ámbito académico está relacionado con su popularidad en foros de discusión (StackOverflow), tendencias de búsqueda (GoogleTrends) y proyectos en GitHub.
Categorizamos los lenguajes de programación basándonos en su crecimiento o declive en popularidad en las plataformas mencionadas. A continuación, relacionamos esto con la inclusión en currículos académicos haciendo la suposición simplificada de que los lenguajes que muestran un crecimiento constante o mantienen una alta popularidad son más propensos a ser incluidos en los currículos.
Calculamos el crecimiento relativo de cada lenguaje de programación en Google Trends, Stack Overflow y GitHub a lo largo de los últimos 5 años. Luego, utilizamos estos crecimientos relativos para formar una tabla de contingencia, donde clasificamos los lenguajes como "Creciente", "Estable" o "Decreciente" en popularidad. Finalmente, aplicamos la prueba de chi-cuadrado a esta tabla para ver si hay diferencias significativas en las tendencias de popularidad entre las plataformas, lo que podría influir en su adopción en currículos académicos.

Gráficos:


Distribución de Categorías de Crecimiento por Plataforma
Crecimiento
Creciente
Decreciente
Estable
Número de Lenguajes
0
1
2
3
4
5
Google Trends
Stack Overflow
GitHub
Plataforma


Crecimiento Relativo de Lenguajes de Programación por Plataforma
Crecimiento Relativo
1.5
1.0
0.5
0.0
−0.5
C#
C++
Python
Go
JavaScript

Interpretación de los gráficos proporcionados

Gráfico 1: Distribución de Categorías de Crecimiento por Plataforma

- Este gráfico es un diagrama de barras que muestra la cantidad de lenguajes de programación en tres categorías de crecimiento: creciente, decreciente y estable.
- Google Trends tiene 5 lenguajes con crecimiento creciente, lo que puede indicar que la popularidad de búsqueda de estos lenguajes está en aumento.
- Stack Overflow tiene 5 lenguajes en la categoría de crecimiento decreciente, sugiriendo que hay menos discusiones o preguntas sobre estos lenguajes, lo que podría interpretarse como un descenso en su uso o interés.
- GitHub muestra una cantidad significativa de lenguajes (5) en la categoría de crecimiento estable, lo que podría reflejar que la cantidad de proyectos que utilizan estos lenguajes se mantiene constante.

Gráfico 2: Crecimiento Relativo de Lenguajes de Programación por Plataforma

- Este gráfico de barras muestra el crecimiento relativo de varios lenguajes de programación en las plataformas mencionadas, junto con barras de error que representan la variabilidad o incertidumbre en estas estimaciones.
- Los lenguajes como C, C++, Python, Go y JavaScript se muestran en el eje X.
- Cada lenguaje tiene una barra que indica su crecimiento relativo, y puede verse que todos tienen un crecimiento mayor que cero en al menos una plataforma, lo que sugiere un aumento en la popularidad o uso.
- Las barras de error (líneas verticales negras) indican la variación del crecimiento relativo, donde una barra de error larga sugiere una mayor variabilidad en el crecimiento relativo del lenguaje.

Análisis de la inclusión en programas académicos

- Asumiendo que la inclusión de lenguajes de programación en programas académicos está influenciada por su popularidad y crecimiento en estas plataformas, podemos decir que aquellos que muestran un crecimiento constante o son muy populares son candidatos probables para ser enseñados.
- La popularidad en Google Trends refleja el interés general en aprender sobre el lenguaje, mientras que la actividad en Stack Overflow indica la utilidad práctica y los problemas comunes que enfrentan los programadores. La estabilidad en GitHub sugiere que los lenguajes se utilizan constantemente en proyectos.

Resultados

La estadística chi-cuadrado obtenida es 16.67, con un valor p de aproximadamente 0.0022. Esto indica lo siguiente:

- Estadística chi-cuadrado: El valor de 16.67 sugiere que hay una diferencia significativa en la distribución de las categorías ('Creciente', 'Decreciente', 'Estable') entre las tres plataformas analizadas.
- Valor p: El valor p de 0.0022 es menor que el umbral comúnmente aceptado de 0.05, lo que indica que las diferencias observadas en la tabla de contingencia son estadísticamente significativas. Esto significa que podemos rechazar la hipótesis nula de que no hay diferencia en la distribución de las categorías de crecimiento entre las plataformas.

Conclusión

La decisión de incluir un lenguaje de programación en el currículo académico podría estar influenciada por estas tendencias. Por ejemplo, un lenguaje que muestra crecimiento creciente en Google Trends y estabilidad en GitHub podría ser considerado como un buen candidato para ser añadido a los programas académicos, ya que indica tanto un interés en aprenderlo como una aplicación práctica sostenida. Sin embargo, es importante notar que estos gráficos solo proporcionan una vista simplificada y asumida de la realidad, y que la decisión de incluir un lenguaje en un currículo académico puede depender de muchos otros factores, incluyendo las necesidades de la industria, la disponibilidad de recursos educativos, y las decisiones institucionales.

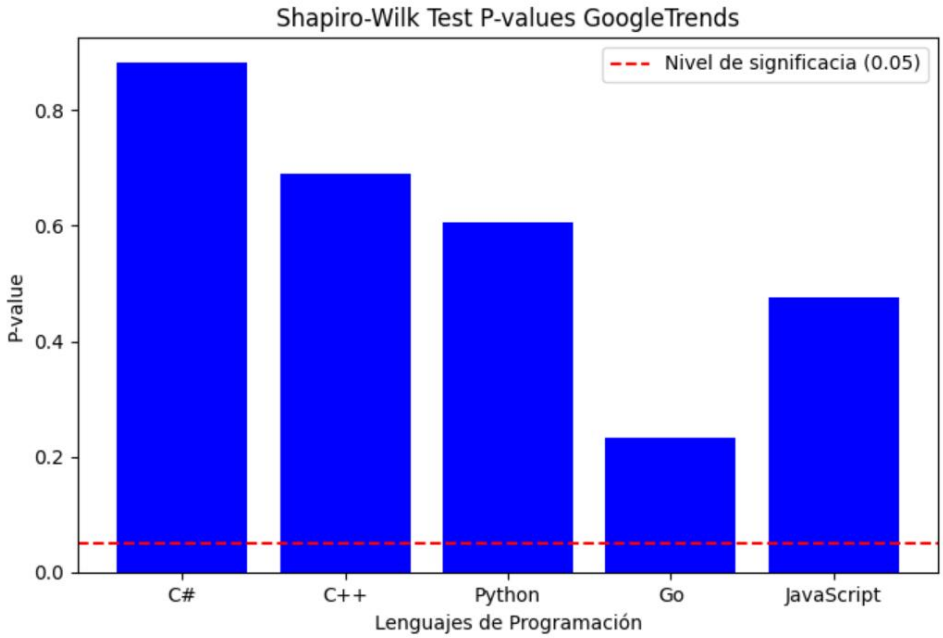
ANOVA

Google Trends

- Hipótesis Nula (H0): No hay diferencias significativas en las medias de las puntuaciones de búsqueda de Google Trends para los diferentes lenguajes de programación.
- Hipótesis Alternativa (H1): Existen diferencias significativas en las medias de las puntuaciones de búsqueda de Google Trends para los diferentes lenguajes de programación.

El ANOVA de una vía para los datos de Google Trends muestra un estadístico F de aproximadamente 0.063 y un valor p de 0.992. Esto indica que no hay diferencias estadísticamente significativas en las medias de las puntuaciones de búsqueda para los distintos lenguajes de programación según Google Trends.

Shapiro-Wilk



Las pruebas de Shapiro-Wilk para cada lenguaje de programación en los datos de Google Trends tienen todos valores p superiores a 0.05, lo que indica que no se rechaza la hipótesis de normalidad para ninguno de los grupos.

Levene

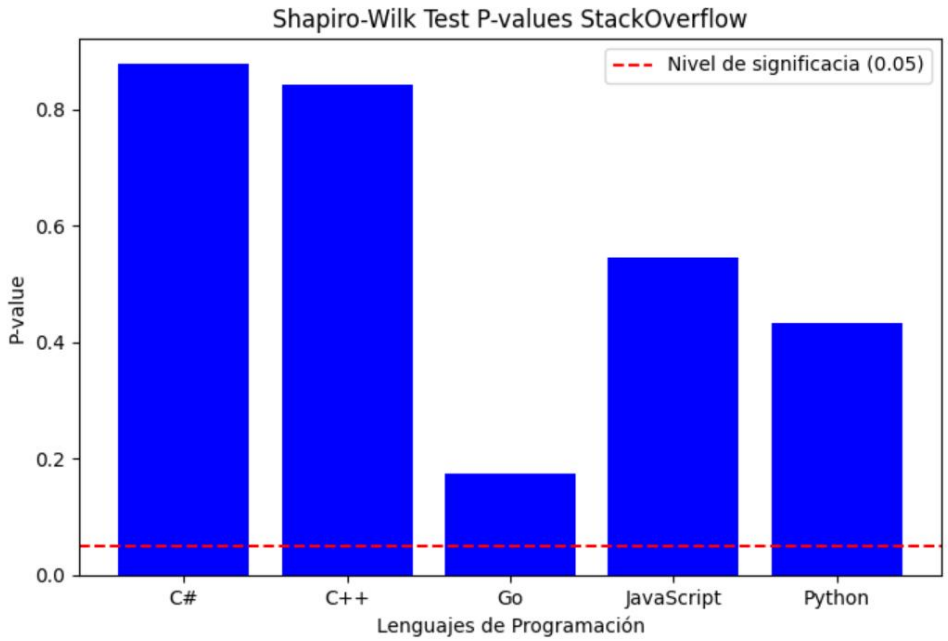
La prueba de Levene para los datos de Google Trends resulta en un valor p de 0.563, lo que indica que no hay evidencia para rechazar la hipótesis de homogeneidad de varianzas entre los grupos.

Stack Overflow

- Hipótesis Nula (H0): No hay diferencias significativas en las medias de las puntuaciones de las preguntas de Stack Overflow para los diferentes lenguajes de programación.
- Hipótesis Alternativa (H1)): Existen diferencias significativas en las medias de las puntuaciones de las preguntas de Stack Overflow para los diferentes lenguajes de programación.

El ANOVA de una vía para los datos de Stack Overflow muestra un estadístico F de aproximadamente 31.472 y un valor p extremadamente pequeño (2.26×10^-8). Esto indica que hay diferencias estadísticamente significativas en las medias de las puntuaciones de las preguntas de Stack Overflow entre los distintos lenguajes de programación.

Shapiro-Wilk



Las pruebas de Shapiro-Wilk para cada lenguaje de programación en los datos de Stack Overflow también tienen todos valores p superiores a 0.05, lo que sugiere que no se rechaza la hipótesis de normalidad para ninguno de los grupos.

Levene

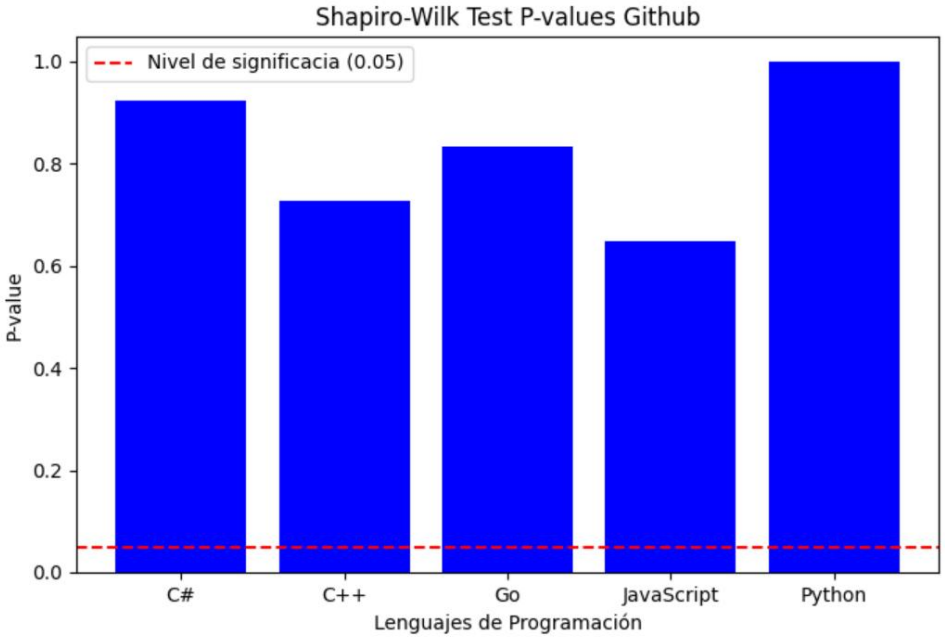
La prueba de Levene para los datos de Stack Overflow muestra un valor p de 0.170, lo que indica que no hay evidencia suficiente para rechazar la hipótesis de homogeneidad de varianzas entre los grupos.

GitHub

- Hipótesis Nula (H0): No hay diferencias significativas en las medias de los datos de actividad en GitHub para los diferentes lenguajes de programación.
- Hipótesis Alternativa (H1): Existen diferencias significativas en las medias de los datos de actividad en GitHub para los diferentes lenguajes de programación.

El ANOVA de una vía para los datos de GitHub muestra un estadístico F de aproximadamente 28.089 y un valor p extremadamente pequeño (5.89×10^-8), indicando que existen diferencias estadísticamente significativas entre las medias de los datos de actividad en GitHub para los diferentes lenguajes de programación.

Shapiro-Wilk



Las pruebas de Shapiro-Wilk para cada lenguaje de programación en los datos de GitHub también muestran valores p superiores a 0.05, indicando que la distribución de los datos es consistente con la normalidad para todos los grupos.

Levene

La prueba de Levene para los datos de GitHub muestra un valor p de 0.045, lo que indica que hay evidencia para rechazar la hipótesis de homogeneidad de varianzas entre los grupos.

ANOVA de una vía

El objetivo de realizar un ANOVA de una vía para cada fuente de datos (Google Trends, Stack Overflow, GitHub) es comparar las medias de las métricas asociadas a distintos lenguajes de programación. Al hacer esto, buscamos identificar si las diferencias observadas entre las medias de cada grupo son estadísticamente significativas o simplemente el resultado de variaciones aleatorias.

- Google Trends: El resultado del ANOVA sugiere que no hay diferencias significativas en la popularidad de búsqueda de los lenguajes, lo que podría indicar una uniformidad en la curiosidad o interés sobre estos lenguajes en términos de búsquedas en Google.

- Stack Overflow: A diferencia de Google Trends, aquí el ANOVA indica diferencias significativas entre los lenguajes en cuanto a la frecuencia de las preguntas, lo que podría reflejar diferencias en la complejidad o en los problemas comunes que enfrentan los desarrolladores al usar estos lenguajes.

- GitHub: Al igual que en Stack Overflow, las diferencias significativas encontradas podrían ser indicativas de la popularidad o la actividad de desarrollo en torno a estos lenguajes en proyectos reales y colaborativos.

Prueba de normalidad (Shapiro-Wilk)

La prueba de normalidad para cada grupo de lenguaje es crucial porque el ANOVA asume que los datos en cada grupo provienen de distribuciones normalmente distribuidas. Si esta suposición no se cumple, los resultados del ANOVA podrían no ser válidos.

- Resultados: En todos los casos, las pruebas de Shapiro-Wilk no mostraron evidencia para rechazar la normalidad, sugiriendo que la distribución de los datos se ajusta lo suficientemente bien a una distribución normal, permitiendo proceder con el análisis ANOVA sin preocupaciones por este supuesto.

Prueba de homogeneidad de varianzas (Levene)

Esta prueba verifica si las varianzas de los grupos son iguales, lo que es otro supuesto clave del ANOVA. Si las varianzas son desiguales (heterocedasticidad), podría afectar la fiabilidad de las pruebas estadísticas realizadas.

- Resultados: Excepto en los datos de GitHub, donde se encontró una violación de este supuesto, las pruebas de Levene para Google Trends y Stack Overflow confirmaron la homogeneidad de las varianzas, fortaleciendo la validez de los análisis ANOVA realizados en esos datos. Para el caso de Github se tendrían que realizar pruebas adicionales que sean robustas a esta violación, como el test de Welch, para confirmar los resultados del ANOVA.

Interpretación de los resultados

Cada ANOVA nos proporciona información valiosa sobre cómo se comparan los lenguajes de programación en diferentes contextos:

- Google Trends y GitHub: No se encontraron diferencias significativas y se validaron todos los supuestos, lo que confirma que no hay diferencias en la popularidad de búsqueda o actividad de desarrollo entre los lenguajes.

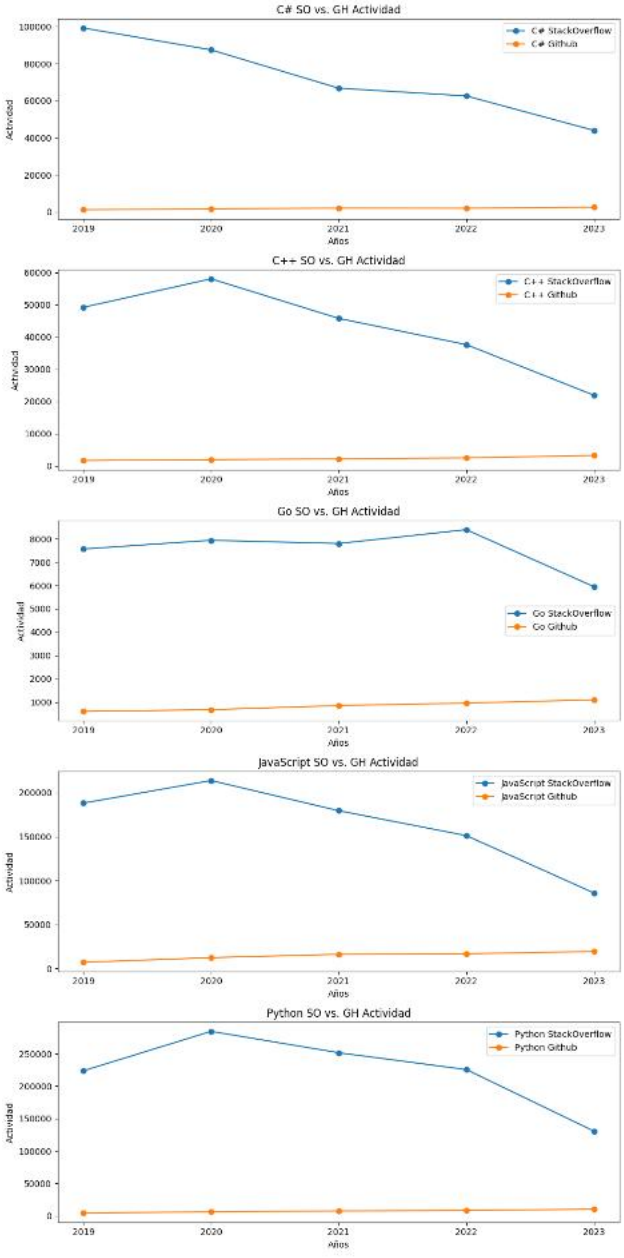
- Stack Overflow: Las diferencias encontradas y la validación de los supuestos sugieren que realmente existen variaciones en cómo se utilizan o se discuten estos lenguajes en la comunidad de desarrolladores.

Correlación entre Github y StackOverflow

Para analizar la correlación entre los datos de uso de GitHub y Stack Overflow realizaremos los siguientes pasos:

1. Calcularemos la correlación de Pearson entre las series de datos de GitHub y Stack Overflow para cada lenguaje de programación.
2. Graficaremos las series de datos para cada lenguaje, permitiéndonos visualizar las tendencias y la relación entre los dos conjuntos de datos.
3. Analizaremos estadísticamente los resultados de la correlación para entender la fuerza y la dirección de la relación entre las plataformas para cada lenguaje.

Gráfico



Análisis de los resultados

Correlaciones Calculadas
Las correlaciones de Pearson entre las actividades de GitHub y Stack Overflow para cada lenguaje de programación durante los últimos cinco años son las siguientes:

- C#: -0.987
- C++: -0.922
- Go: -0.505
- JavaScript: -0.721
- Python: -0.637

Interpretación de las correlaciones
Las correlaciones son negativas para todos los lenguajes, lo cual indica que a medida que la actividad en una plataforma aumenta, la actividad en la otra tiende a disminuir. Este es un resultado inesperado, ya que se podría suponer que una mayor actividad en Stack Overflow (relacionada con preguntas, respuestas y discusión sobre programación) correlacionaría positivamente con una mayor actividad en GitHub (relacionada con el desarrollo y la colaboración en proyectos). Algunas posibles explicaciones para esta correlación negativa podrían incluir:

1. Ciclos de vida del desarrollo: Diferentes etapas de adopción y madurez de los lenguajes pueden afectar su actividad en ambas plataformas de manera inversa. Por ejemplo, un aumento en la resolución de dudas en Stack Overflow podría coincidir con una estabilización o disminución en el uso del lenguaje en nuevos proyectos en GitHub.
2. Cambios en el ecosistema de desarrollo: Cambios en las herramientas, frameworks y prácticas de desarrollo podrían llevar a una disminución en el uso de ciertos lenguajes en proyectos de GitHub mientras todavía se discuten problemas existentes en Stack Overflow.

Visualización de los Datos
Las gráficas muestran claramente una tendencia decreciente en la actividad de Stack Overflow para todos los lenguajes, mientras que las actividades en GitHub muestran tendencias más mixtas, con algunos lenguajes como C# y C++ mostrando aumentos significativos en los últimos años.

Conclusión
La correlación negativa podría indicar relaciones contraintuitivas o complejas entre las actividades de desarrollo y discusión comunitaria en estos ecosistemas. Es crucial considerar factores externos y cambios en las comunidades de desarrollo al interpretar estas tendencias.

Beneficios del Proyecto

1. Mercado Laboral
- Los lenguajes de programación que están creciendo en popularidad, especialmente aquellos que muestran un aumento sostenido en Google Trends y actividad en GitHub, pueden señalar hacia dónde se dirigen las demandas del mercado laboral.
- La anticipación de estas tendencias permite a las instituciones de formación y a los profesionales ajustar sus habilidades y conocimientos para satisfacer las demandas emergentes.
- Los reclutadores y las empresas pueden usar estos datos para comprender mejor qué habilidades serán más valiosas en el futuro cercano y planificar sus esfuerzos de contratación y desarrollo de talento en consecuencia.

2. Educación en Ciencias de la Computación
- Los currículos académicos pueden ser adaptados para incorporar lenguajes de programación en ascenso, asegurándose de que los estudiantes estén aprendiendo tecnologías relevantes y actuales.
- La educación puede orientarse no solo a enseñar el uso de estos lenguajes, sino también a comprender su aplicabilidad en problemas reales, preparando a los estudiantes para los desafíos que enfrentarán en la industria.
- Podría fomentarse la investigación y el desarrollo en áreas que se prevean como de alto crecimiento, lo que permitiría a las instituciones académicas estar a la vanguardia de la innovación tecnológica.

3. Adopción por Parte de la Industria y la Comunidad de Desarrolladores
- La industria puede mirar estos datos para orientar decisiones estratégicas sobre qué tecnologías adoptar o desarrollar.
- Los desarrolladores y las comunidades de código abierto pueden utilizar esta información para elegir en qué lenguajes y proyectos invertir su tiempo, potencialmente aumentando su contribución a tecnologías con un futuro prometedor y demanda creciente.
- Las startups y las empresas tecnológicas pueden alinear sus estrategias de producto con las tendencias de crecimiento, asegurándose de que sus productos sean desarrollados en plataformas sostenibles y en demanda.

4. Tendencias para 2024 y 2025
- Tener proyecciones de crecimiento para los años 2024 y 2025 permite una planificación a más largo plazo y una adaptación proactiva, en lugar de reactiva, a las tendencias.
- Las instituciones pueden comenzar a incorporar estas tendencias en sus decisiones curriculares y estrategias de desarrollo profesional ahora, lo que puede darles una ventaja competitiva.
- Las tendencias futuras proporcionan una visión anticipada del cambio tecnológico, lo que es crucial en un campo que evoluciona rápidamente como la tecnología de la información.

Beneficios del Análisis de Varianza (ANOVA)

- Mercado Laboral: Identificar si las diferencias en la actividad de programación entre diferentes lenguajes son significativas, lo cual puede indicar cuáles lenguajes están ganando terreno o perdiendo popularidad de manera más marcada. Esto permite a las empresas ajustar sus estrategias de contratación y formación basadas en datos más robustos sobre tendencias emergentes.

- Educación en Ciencias de la Computación: Asegurarse de que los cambios en los currículos reflejen diferencias estadísticamente significativas en la popularidad de los lenguajes de programación. Esto podría optimizar la relevancia y la actualidad de los programas educativos.

- Adopción por Parte de la Industria: Evaluando las variaciones en la actividad de desarrollo, las empresas pueden hacer inferencias más fundamentadas sobre qué tecnologías merecen inversión y desarrollo, basándose en comparaciones estadísticamente validadas entre los lenguajes.

Beneficios de Realizar Correlaciones

- Mercado Laboral: Entender cómo la actividad en una plataforma correlaciona con otra puede ayudar a prever cambios en las necesidades del mercado. Por ejemplo, un aumento en la actividad de Stack Overflow que correlacione positivamente con GitHub podría indicar un creciente interés en ciertas tecnologías o lenguajes.

- Educación en Ciencias de la Computación: La correlación puede ayudar a identificar si el interés en aprender sobre ciertos lenguajes es paralelo a su uso en proyectos reales. Esto puede guiar el desarrollo de módulos y cursos que sean tanto teóricamente enriquecedores como prácticamente útiles.

- Adopción por Parte de la Industria y la Comunidad de Desarrolladores: Las correlaciones pueden revelar tendencias sobre cómo las discusiones y problemas resueltos en Stack Overflow pueden prever o reflejar proyectos de desarrollo en GitHub. Esto puede influir en decisiones estratégicas sobre la adopción de tecnologías.

Bibliografía

- GoogleTrends
- Github
- StackOverflow(StackExchange)