

Herramientas para Proyectos de Ciencia de Datos



trabajo.

on y
del

Comunidad

del

Facilidad de Uso

10.0

Herramientas para Proyectos de Ciencia de Datos

Bloque F: Dominio de las herramientas y lenguajes esenciales para ejecutar proyectos de ciencia de datos con eficiencia y profesionalismo

Big Data

Visualización

ra:

jes y
culada
típico.

Python R

Comparación de lenguajes para ciencia de
datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Saberes Declarativos:

- Preparación del ambiente de trabajo.
- Plataformas para colaboración y control de versiones.
- Lenguajes de programación del científico de datos.

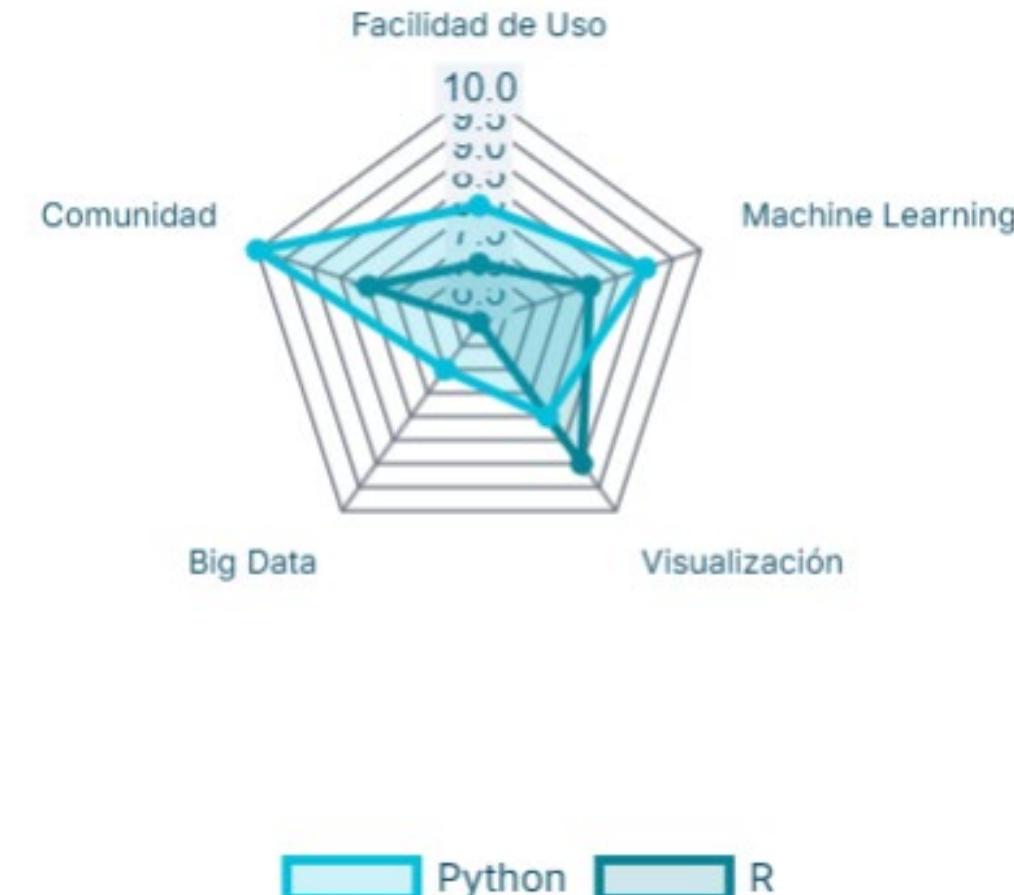
Evidencia y Ponderación:

10%



Tabla comparativa:

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.



Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Objetivos de Aprendizaje del Bloque

- Preparación del ambiente de trabajo.

Facilidad de Uso
100%

Saberes Declarativos

Comprender la importancia de la preparación del ambiente de trabajo, las plataformas de colaboración como Git y GitHub, y los lenguajes de programación clave para la ciencia de datos: Python y R.

Saberes Procedimentales

Ser capaz de configurar un ambiente de trabajo profesional, utilizar sistemas de control de versiones de manera efectiva, y diferenciar las características y aplicaciones de los principales lenguajes de programación.

Evidencia y Ponderación:

10%

Tabla comparativa:

Sobre el uso de diferentes lenguajes y tecnologías, incluida la arquitectura, alrededor del problema prototípico.

Durante las semanas 11 y 12, desarrollaremos las competencias técnicas fundamentales que todo científico de datos debe dominar en el entorno profesional actual.

Big Data

Visualización



Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Saberes Declarativos:

Estructura del Bloque F

- Plataformas para colaboración y control de versiones.

01

Preparación del Ambiente

Configuración de entornos virtuales, gestión de dependencias y mejores prácticas para un espacio de trabajo organizado

02

Colaboración y Control de Versiones

Dominio de Git y GitHub para trabajo en equipo, seguimiento de cambios y gestión profesional de proyectos

Facilidad de Uso

10.0

03

Lenguajes de Programación

Python y R como herramientas fundamentales, sus ecosistemas, fortalezas y aplicaciones específicas en ciencia de datos

Tabla comparativa:

10%

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.

Python R

Comparación de lenguajes para ciencia de datos.

Resumen E: Herramientas

¿Por qué es Fundamental la Preparación del Ambiente?

Saberes Declarativos:

- Preparación del ambiente de trabajo

Un ambiente de trabajo bien configurado es la base de la productividad en ciencia de datos.

La correcta preparación evita conflictos de dependencias, facilita la reproducibilidad de análisis y permite un desarrollo más eficiente.

10%

Tabla comparativa:

Sobre el uso de diferentes lenguajes y arquitecturas, vinculados al Problema Prototípico

The screenshot shows the Anaconda Cloud interface. At the top, there's a navigation bar with icons for IA UNIR, Home, Cloud: Data Science and Python Tools from Anaconda, and other links like Laureate cursos, Exámenes online, and Banner UNITEC. A search bar at the top right says "Search Anaconda packages, collections, documentation, and more!" with a "CMD + K" keyboard shortcut. On the left, a sidebar has a teal header with a "+ Create" button and a list of links: Dashboard (highlighted in teal), Resources, Getting Started, My Learning, Documentation, Forum, and Support. The main content area is titled "Welcome Back, Monica Guadalupe!". It features three buttons: "Launch a Notebook" (with a Jupyter icon), "Take a Course" (with a graduation cap icon), and "Join the Community" (with a people icon). Below this, there's a section titled "Explore Anaconda" with five cards: "Notebooks" (jupyter logo), "Learning" (graduation cap logo), "Forum" (globe logo), "Docs" (document logo), and "AI Navigator" (brain logo). At the bottom, there are five more cards: "Data Wrangler" (green globe), "Data Cleaning" (black circle), "Data Modeling" (blue M), "Data Visualization" (light blue square), and "Data Science" (green globe).

Bloque E: Herramientas

Preparación del Ambiente

¿Por qué es Fundamental la Preparación del Ambiente?

Saberes Declarativos:

- Preparación del ambiente de trabajo
- Plataformas para colaboración y

Los entornos virtuales aíslan los proyectos, asegurando que cada uno tenga las versiones exactas de las librerías que necesita sin interferir con otros trabajos.

Evidencia y Ponderación:

Tabla comparativa:

10%

Sobre el uso de diferentes lenguajes y arquitecturas, vinculados al Problema Prototípico



- Preparación del ambiente de trabajo.
- Plataformas para colaboración y control de versiones.

Entornos Virtuales: Aislamiento y Control

Aislamiento de Proyectos

Cada proyecto tiene su propio conjunto de dependencias sin conflictos. Un proyecto puede usar Pandas 1.3 mientras otro usa Pandas 2.0 sin problemas.

Reproducibilidad Garantizada

Los entornos virtuales permiten documentar y recrear exactamente las mismas condiciones de desarrollo, fundamental para compartir código y resultados.

Gestión Eficiente

Facilita la instalación, actualización y eliminación de paquetes sin afectar el sistema operativo o otros proyectos en desarrollo.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Herramientas para Gestión de Ambientes

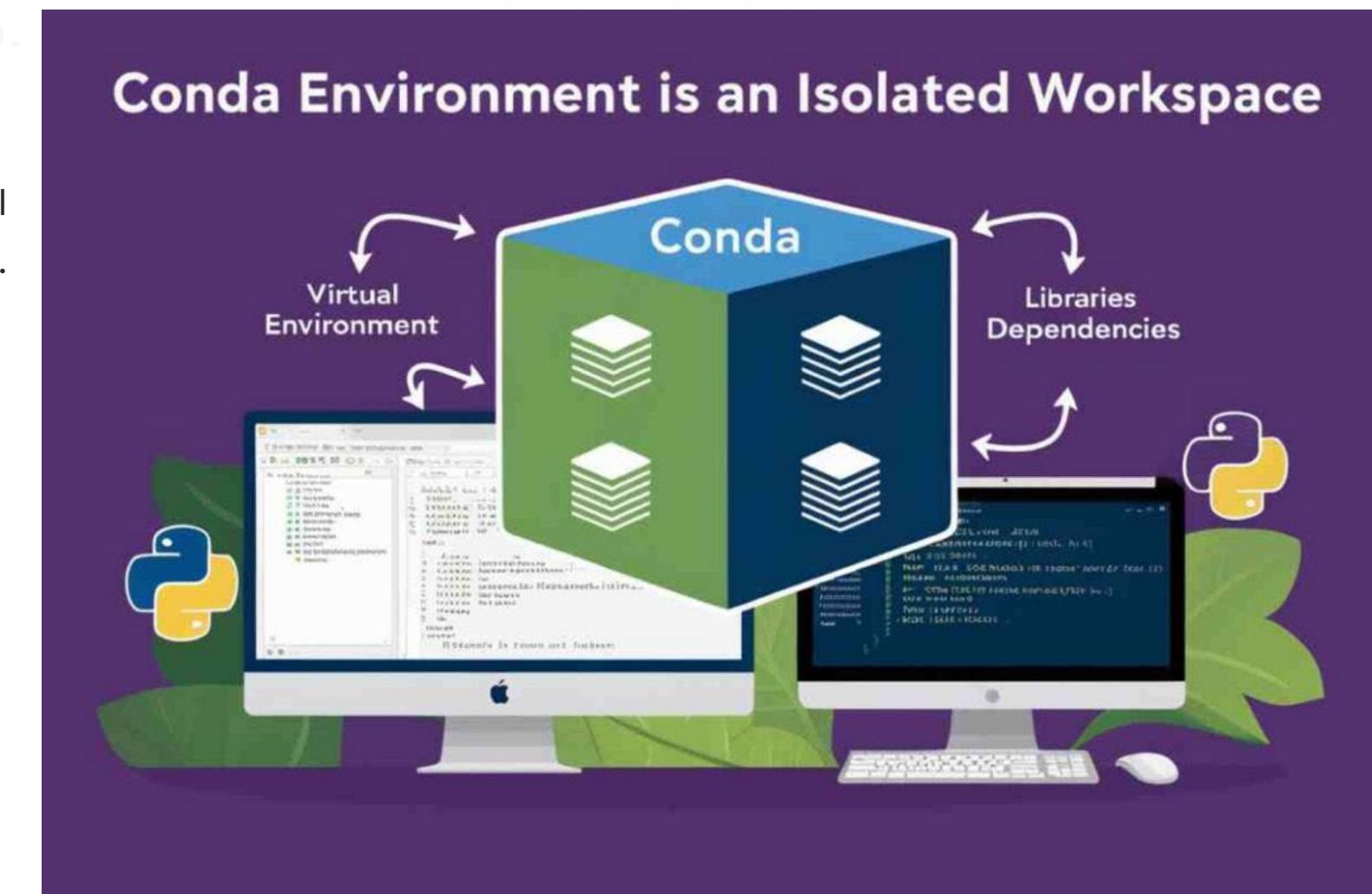
Saberes Declarativos:

Conda

• Plataformas para colaboración y
compartir versiones
Un gestor de paquetes y entornos multiplataforma que maneja tanto paquetes de Python como binarios de otros lenguajes. Ideal para proyectos de ciencia de datos con dependencias complejas.

Ventajas clave:

- Gestión simplificada de dependencias complejas
- Compatible con múltiples lenguajes
- Excelente para bibliotecas científicas
- Integración con Anaconda



Bloque F: Herramientas para Proyectos de Ciencia de Datos

Herramientas para Gestión de Ambientes

Saberes Declarativos:

Instalación

- Plataformas para colaboración y

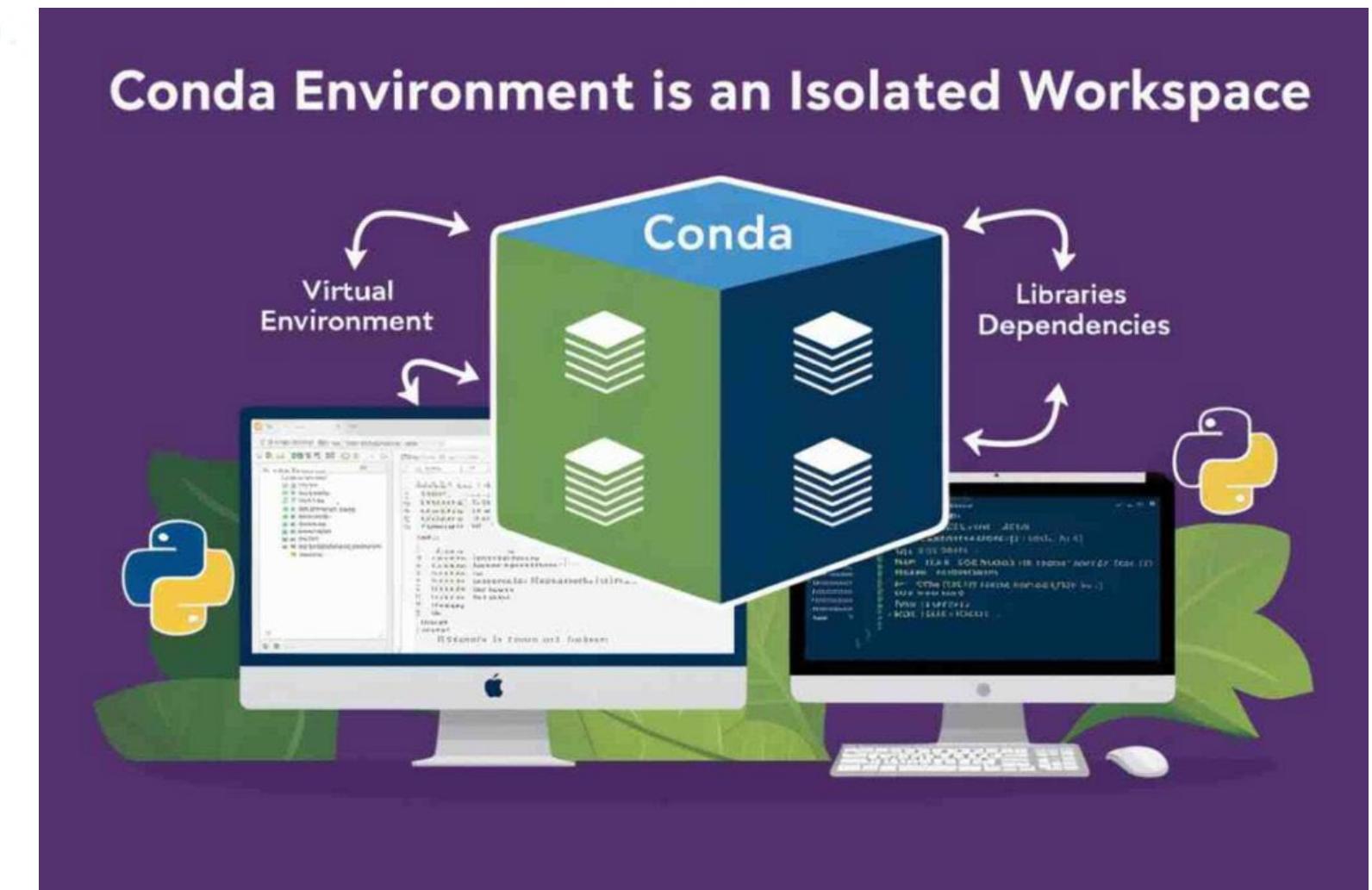
[Installing conda — conda 25.9.2.dev61 documentation](#)

- Lenguajes de programación del científico de datos.

Evidencia y Ponderación:

Tabla comparativa:

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.



datos.

Navigation

User guide

Getting started with conda

Installing conda

Installing on Windows

Installing on macOS

Installing on Linux

RPM and Debian Repositories for
Miniconda

Tasks

Configuration

Concepts

Troubleshooting

Cheatsheet

Configuration

Commands

Release notes

Glossary

Developer guide

 > User guide > Installing conda

Installing conda

To install conda, you must first pick the right installer for you. The following are the most popular installers currently available:

[Miniconda](#)

Miniconda is a minimal installer provided by Anaconda. Use this installer if you want to install most packages yourself.

[Anaconda Distribution](#)

Anaconda Distribution is a full featured installer that comes with a suite of packages for data science, as well as Anaconda Navigator, a GUI application for working with conda environments.

[Miniforge](#)

Miniforge is an installer maintained by the conda-forge community that comes preconfigured for use with the conda-forge channel. To learn more about conda-forge, visit [their website](#).

Note

Miniconda and Anaconda Distribution come preconfigured to use the [Anaconda](#)

 On this page

[System requirements](#)

[Regular installation](#)

[Installing in silent mode](#)

[Cryptographic hash verification](#)

 [Edit on GitHub](#)

 [Show Source](#)



Bloque F: Herramientas para Proyectos de Ciencia de Datos

Herramientas para Gestión de Ambientes

Saberes Declarativos:

Preparación del ambiente de trabajo.

• Plataformas para colaboración y control de versiones.

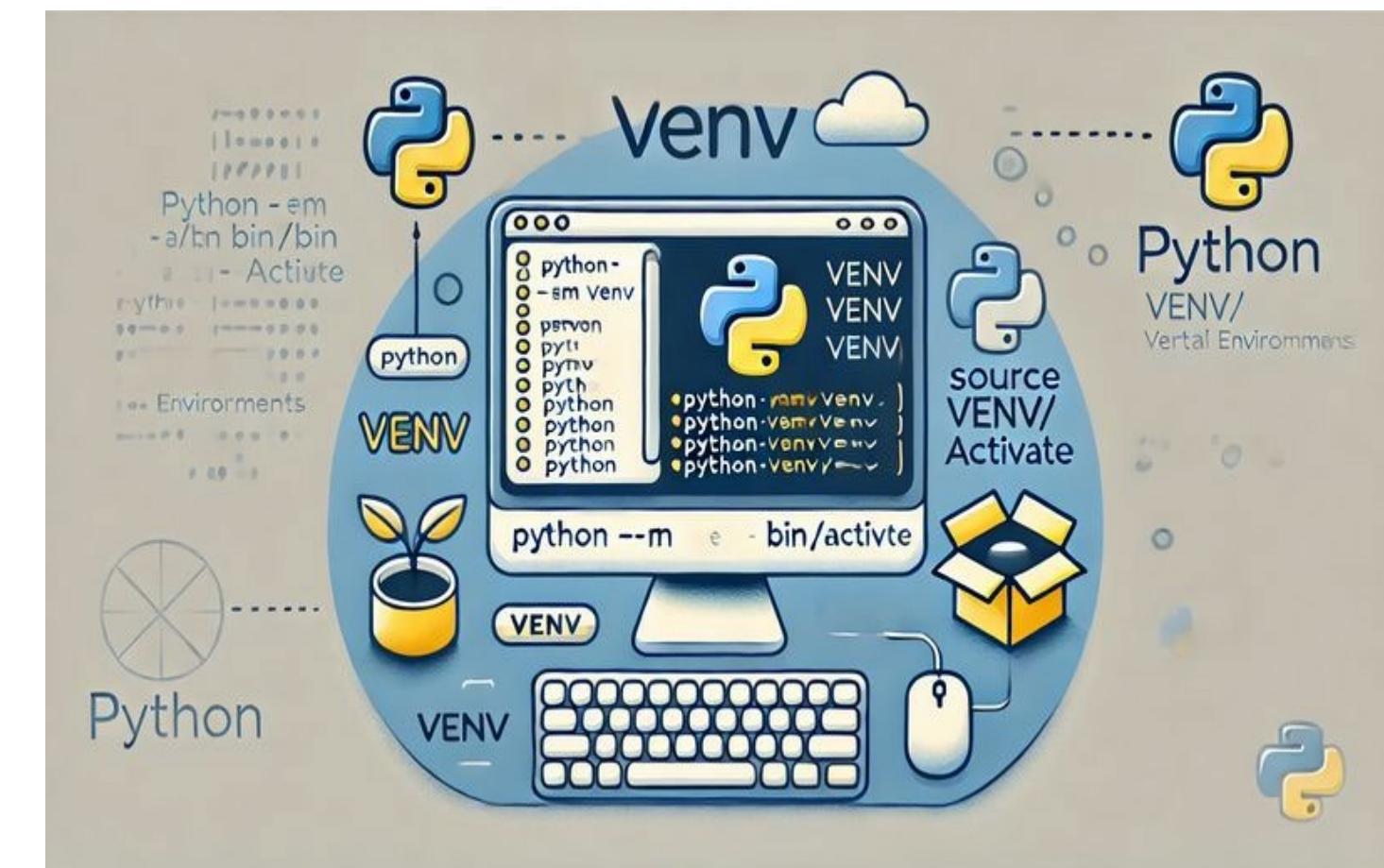
La herramienta nativa de Python para crear entornos virtuales. Ligera, rápida y perfecta para proyectos que solo requieren paquetes de Python.

Ventajas clave:

- Incluida en Python 3.3+
- Sin instalaciones adicionales
- Rápida y ligera
- Ideal para proyectos simples

Evidencia y ponderación:

Tabla comparativa:
Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.



Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Herramientas para Gestión de Ambientes

Saberes Declarativos:

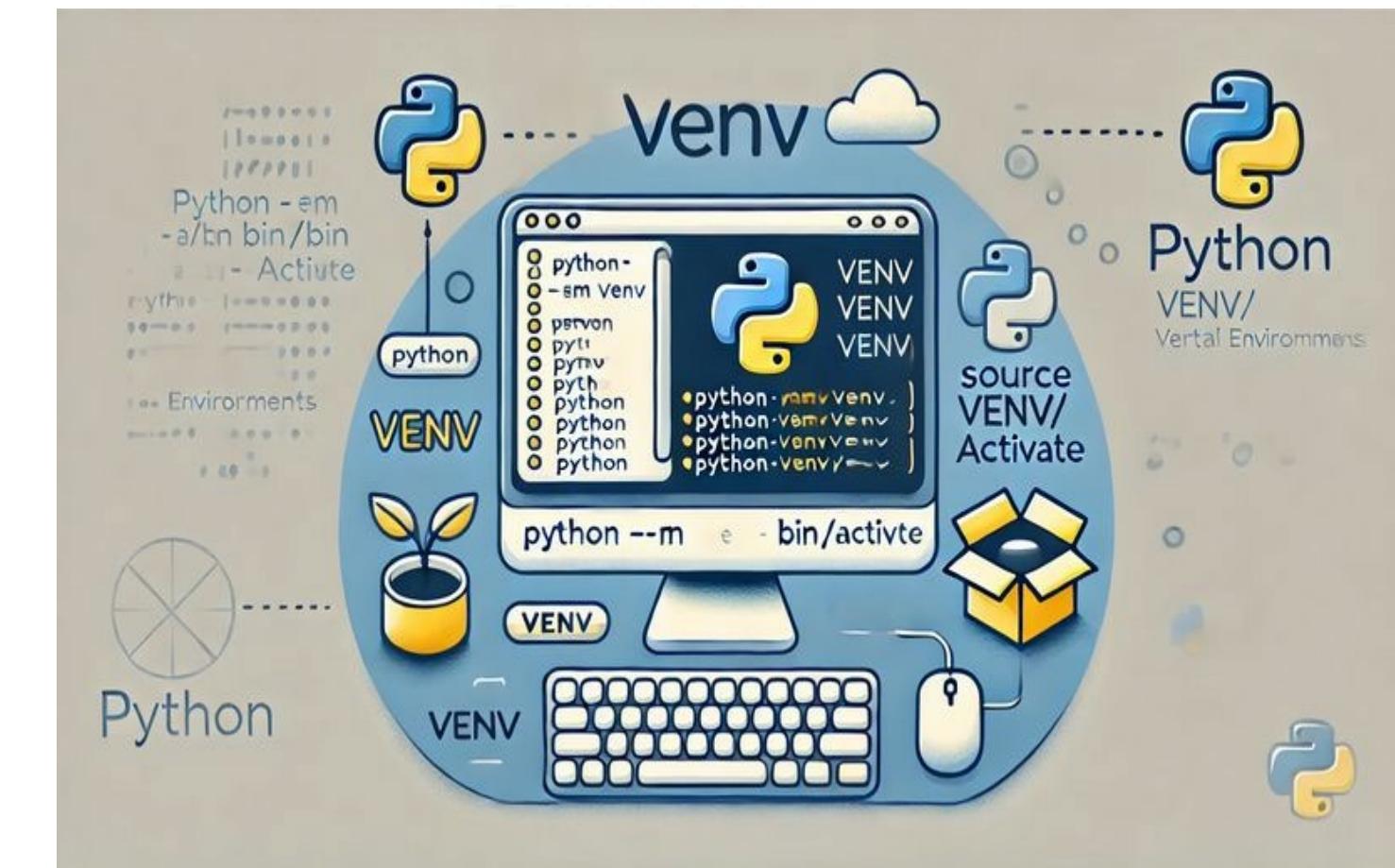
Instalación

- Plataformas para colaboración y control de versiones
- venv — Creación de entornos virtuales — documentación de Python - 3.8.20

Evidencia y Ponderación:

Tabla comparativa:

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.



Comparación de lenguajes para ciencia de datos.

https://docs.python.org/es/3.8/library/venv.html

IA UNIR | Laureate cursos | octorado | OLIVARES ROBLES... | Exámenes online | iti | Exámenes online | BANNER UNITEC | Hacienda cfdi | Code.org | Turnitin | calendario | Otros favoritos

Python » Spanish 3.8.20 3.8.20 Documentation » La Biblioteca Estándar de Python » Software Packaging and Distribution » Búsqueda rápida | Ir a | anterior | siguiente | módulos | índice

Tabla de contenido

- venv — Creación de entornos virtuales
 - Creación de entornos virtuales
 - API
 - Un ejemplo de la extensión de EnvBuilder

Tema anterior

[ensurepip](#) — Ejecutando el instalador pip

Próximo tema

[zipapp](#) — Gestiona archivadores zip ejecutables de Python

Esta página

Reporta un Bug
Mostrar el código

venv — Creación de entornos virtuales

Nuevo en la versión 3.3.

Código fuente: [Lib/venv/](#)

El módulo `venv` proporciona soporte para crear «entornos virtuales» ligeros con sus propios directorios de ubicación, aislados opcionalmente de los directorios de ubicación del sistema. Cada entorno virtual tiene su propio binario Python (que coincide con la versión del binario que se utilizó para crear este entorno) y puede tener su propio conjunto independiente de paquetes Python instalados en sus directorios de ubicación.

Ver [PEP 405](#) para más información sobre los entornos virtuales de Python.

Ver también: [Python Packaging User Guide: Creating and using virtual environments](#)

Creación de entornos virtuales

La creación de [entornos virtuales](#) se hace ejecutando el comando `venv`:

```
python3 -m venv /path/to/new/virtual/environment
```

Al ejecutar este comando se crea el directorio de destino (creando cualquier directorio padre que todavía no exista) y coloca un archivo `pyvenv.cfg` en él con una clave `home` apuntando a la instalación Python desde la que el comando se lanzó (un nombre común para el directorio destino es `.venv`). También crea un subdirectorio `bin` (o `Scripts` en Windows) contenido una copia/enlace simbólico del binario/s Python (según lo apropiado para la plataforma o los argumentos usados en el momento de la creación del entorno). También crea un directorio (inicialmente vacío) `lib/pythonX.Y/site-packages` (en Windows, este es `Lib\site-packages`). Si se

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Pasos para Configurar tu Ambiente



Contenidos Declarativos:

- Preparación del ambiente de trabajo.
- Instalación de Python
- Descarga e instala la versión más reciente de Python desde python.org o instala Anaconda para un paquete completo con herramientas de ciencia de datos.



Activación y Ponderación:

Activación del Entorno

Activa el entorno con `source nombre_entorno/bin/activate` (Linux/Mac) o `nombre_entorno\Scripts\activate` (Windows).



Creación del Entorno Virtual

Usa `python -m venv nombre_entorno` o `conda create -n nombre_entorno python=3.9` para crear un entorno aislado para tu proyecto.



Big Data

Instalación de Paquetes

Instala las librerías necesarias con `pip install pandas numpy scikit-learn` o `conda install pandas numpy scikit-learn`.



Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

El archivo `requirements.txt` documenta todas las librerías y versiones específicas que tu proyecto necesita. Esto es fundamental para la reproducibilidad.

Preparación del ambiente de trabajo.

Para generar el archivo:

- Plataformas para colaboración y

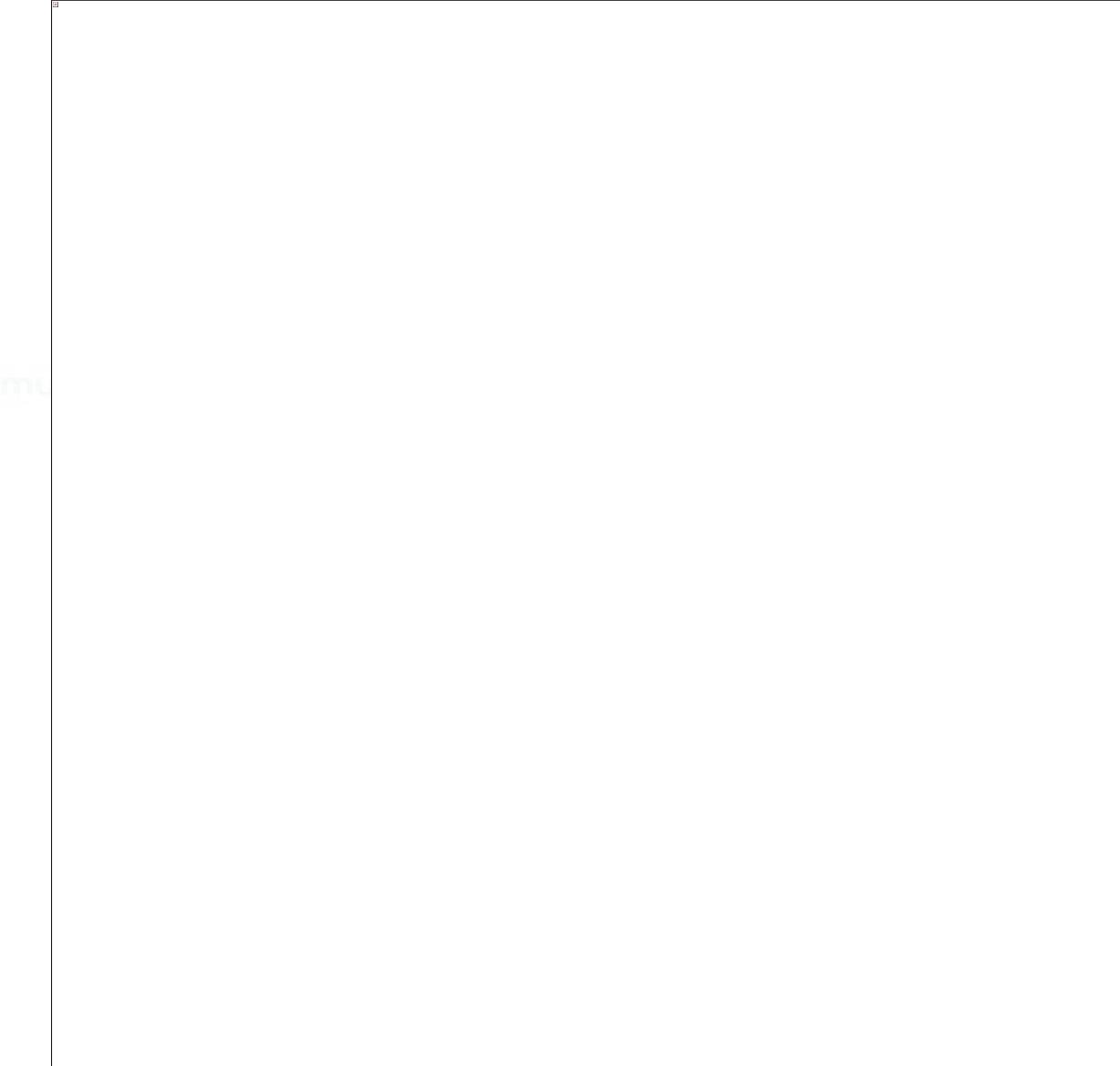
```
pip freeze > requirements.txt
```

- Lenguajes de programación del

Para instalar desde el archivo:

```
pip install -r requirements.txt
```

Esta práctica garantiza que cualquier persona pueda recrear exactamente tu ambiente de desarrollo, facilitando la colaboración y el despliegue de proyectos.



Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Mejores Prácticas de Configuración

Preparación del ambiente de trabajo.

Un entorno por proyecto

Nunca compartas entornos entre proyectos. Cada proyecto debe tener su propio ambiente virtual aislado.

Ciudadano de datos.

Documentación actualizada

Mantén tu `requirements.txt` o `environment.yml` actualizado cada vez que agregues o actualices paquetes.

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.

Facilidad de Uso

Nombres descriptivos

Usa nombres claros para tus entornos que indiquen el proyecto o propósito: `projeto_ventas_2024` en lugar de `env1`.

Comparación de lenguajes para ciencia de datos.

Control de versiones

Incluye archivos de dependencias en tu repositorio Git, pero nunca incluyas la carpeta del entorno virtual en sí.



Python



R

Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Saberes Declarativos:

- Preparación del ambiente de trabajo.
- Plataformas (Jupyter Notebook, Google Colab, etc.)
- Control de versiones (Git, GitHub)
- Lenguajes de programación del científico de datos.

Siguiente sesión

Evidencia y Ponderación:

10%



Tabla comparativa:

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.



Python R

Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Datos

Saberes Declarativos:

Git: Control de Versiones Distribuido

Git es el sistema de control de versiones más utilizado en el mundo del desarrollo de software y la ciencia de datos. Permite rastrear cambios en el código, colaborar con otros desarrolladores y mantener un historial completo del proyecto.

A diferencia de sistemas centralizados, Git es **distribuido**: cada desarrollador tiene una copia completa del repositorio, lo que permite trabajar offline y facilita la colaboración sin depender de un servidor central.

Tabla comparativa:

10%

Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.

Compa

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Conceptos Fundamentales de Git

Saberes Declarativos:

Repository

- Preparación del ambiente de trabajo.
- Plataformas para colaboración y control de versiones.
- Entornos de desarrollo y ejecución.
- Integración continua y entrega continua.
- Documentación y administración de cambios.
- Configuración y personalización.
- Entrenamiento y validación de modelos.
- Visualización y exploración de datos.
- Integración con herramientas de machine learning.
- Uso de lenguajes y frameworks.
- Prácticas de desarrollo y gestión de proyectos.



Evidencia y Ponderación:

Branch (Rama)

Una línea independiente de desarrollo. Permite trabajar en nuevas características sin afectar la rama principal, facilitando la experimentación segura.

Comparación de lenguajes para ciencia de datos.



Commit

Una instantánea de los cambios en el proyecto. Cada commit tiene un mensaje descriptivo y un identificador único que permite rastrear la evolución del código.



Merge

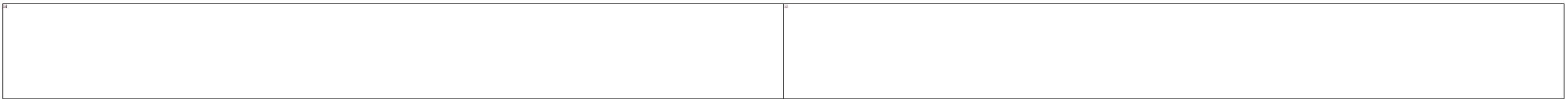
El proceso de integrar cambios de una rama a otra. Combina el trabajo de diferentes desarrolladores o características en una sola versión unificada.

Python R

Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Flujo de Trabajo Básico con Git



- Preparación del ambiente de trabajo.
- Modificar
- Realizas cambios en los archivos de tu proyecto: código, documentación, datos, etc.



Confirmar (Commit)

Guardas los cambios en el historial con un mensaje descriptivo usando `git commit`.

Este ciclo se repite continuamente durante el desarrollo del proyecto, creando un historial detallado y rastreable de todos los cambios realizados.

Agregar (Stage)

Seleccionas qué cambios quieres incluir en el próximo commit con `git add`.

Enviar (Push)

Subes los commits al repositorio remoto con `git push` para compartir tu trabajo.

Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Comandos Git Esenciales

Configuración Inicial

```
git config --global user.name "Tu Nombre"  
git config --global user.email "tu@email.com"  
git init
```

Control de versiones.

Trabajo Diario

- Lenguajes de programación del

```
git status  
git add archivo.py  
git add .  
git commit -m  
"Mensaje descriptivo"
```

10%

Sobre el uso de

Colaboración

```
git clone url_repositorio  
git pull  
git push origin main
```

Comunidad Ramas

```
git branch nombre_rama  
git checkout nombre_rama  
git merge nombre_rama
```

Machine Learning

- Tip profesional:** Escribe mensajes de commit claros y descriptivos. En lugar de "arreglos", usa "Corrige validación de datos en función process_data()".

Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Datos

Saberes Declarativos:

GitHub: Colaboración en la Nube

GitHub es una plataforma web que aloja repositorios Git en la nube, facilitando la colaboración entre equipos distribuidos geográficamente. Va más allá del simple almacenamiento: ofrece herramientas para revisión de código, gestión de proyectos, automatización y documentación.

Con más de 100 millones de usuarios, GitHub se ha convertido en el estándar de facto para el desarrollo colaborativo, siendo esencial en el portafolio profesional de cualquier científico de datos.

Tabla comparativa:

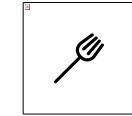
10%



Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.

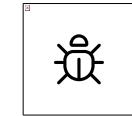
Compa

Características Clave de GitHub



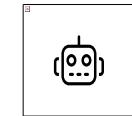
Repositories Remotos

Almacenamiento seguro en la nube de tu código con respaldo automático y acceso desde cualquier lugar.



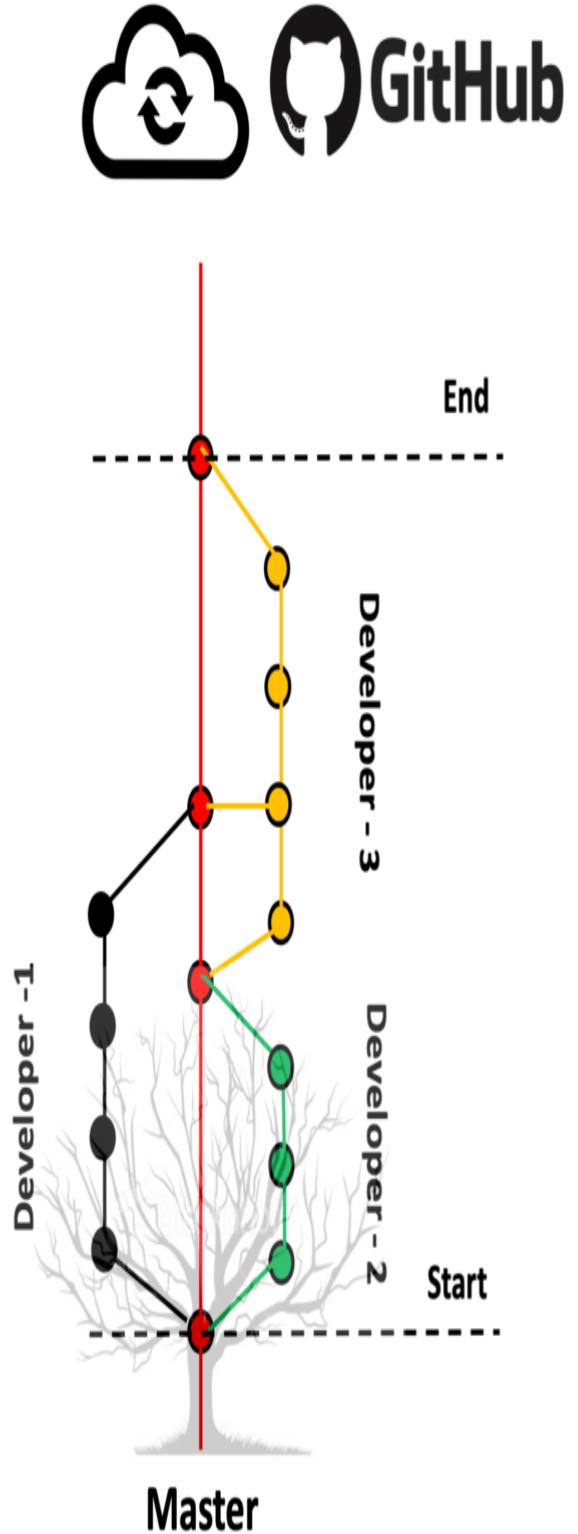
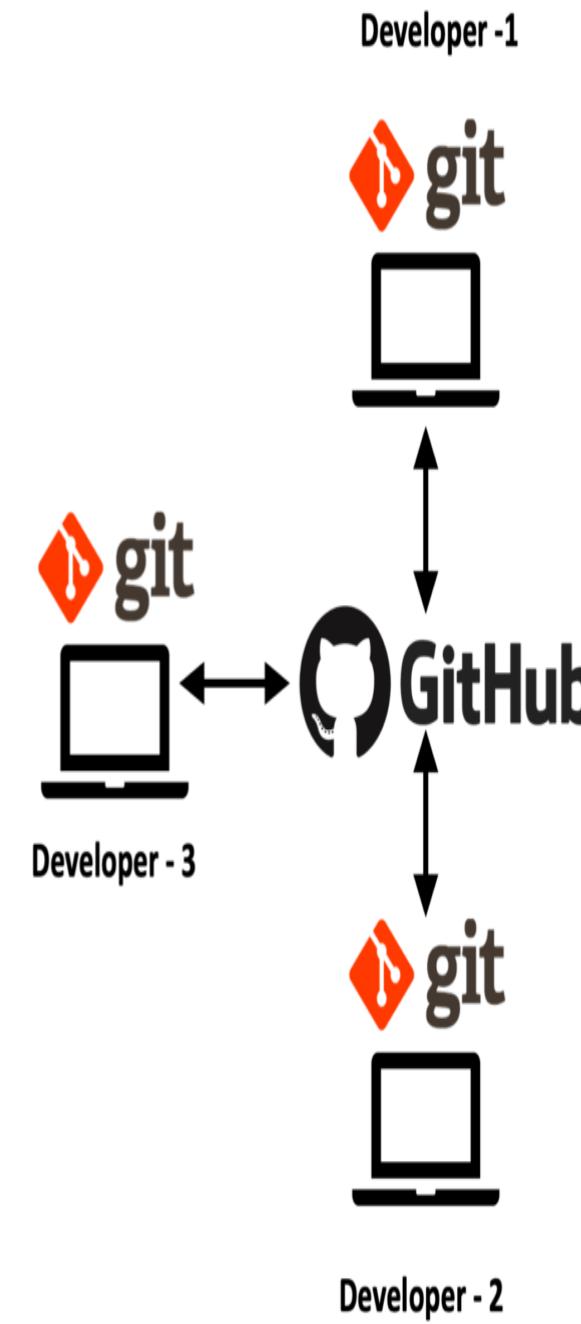
Issues

Sistema de seguimiento de tareas, errores y mejoras que facilita la gestión organizada del proyecto.



Actions

Automatización de pruebas, despliegues y flujos de trabajo mediante integración continua.



Características Clave de GitHub



Pull Requests

Propuesta de cambios que permite revisión de código antes de integrar modificaciones a la rama principal.



Colaboración

Múltiples usuarios pueden trabajar simultáneamente con control de permisos y flujos de aprobación.



Documentación

Wikis y archivos README para documentar el proyecto de forma accesible y profesional.

Current Repository: desktop

Current Branch: the-end-of... #15640 ✓

Pull origin: Last fetched 5 minutes ago 1 ↓

Changes 3 History app/src/ui/diff/seamless-diff-switcher.tsx

3 changed files

app/src/ui/.../seamless-diff-switcher.tsx

app/src/ui/diff/side-by-side-diff.tsx

app/src/ui/diff/text-diff.tsx

```
@@ -19,6 +19,7 @@ import {  
    import { Loading } from '../lib/loading'  
    import { getFileContents, IFileContents } from './syntax-highlighting'  
    import { getTextDiffWithBottomDummyHunk } from './text-diff-expansion'  
} from './text-diff-expansion'  
+ import { textDiffEquals } from './diff-helpers'  
22  
23  
/**  
 * The time (in milliseconds) we allow when loading a diff before  
hen loading a diff before  
@@ -127,7 +128,7 @@ function isSameDiff(prevDiff: IDiff, newDiff: IDiff) {  
    prevDiff === newDiff ||  
    (isTextDiff(prevDiff) &&  
     isTextDiff(newDiff) &&  
     - prevDiff.text === newDiff.text) 130 131 + textDiffEquals(prevDiff, newDiff)  
    )  
}
```

Stashed Changes

Rerender diff when newlines are added

Description

Co-Authors @tidy-dev @sergiou87

Commit to the-end-of-it-all

Flujo de Trabajo Colaborativo en GitHub

Datos

Fork del Repositorio
Crees una copia personal del proyecto en tu cuenta de GitHub para trabajar de forma independiente.

Saberes Declarativos:

- Preparación del ambiente de trabajo.
- Plataformas para colaboración y control de versiones.
- Lenguajes de programación del científico de datos.

Crear Rama
Generas una nueva rama para tu característica o corrección con `git checkout -b feature-nueva`.

Push a GitHub
Subes tu rama al repositorio remoto con `git push origin feature-nueva`.

Tabla comparativa:

Sobre el uso de diferentes lenguajes para la resolución de problemas.

Revisión y Merge
Después de la aprobación, los cambios se integran a la rama principal del proyecto.



Comunidad

Clone Local
Descargas el repositorio a tu computadora con `git clone` para comenzar el desarrollo.

Desarrollo y Commits

Trabajas en los cambios y realizas commits regulares con mensajes descriptivos.

Pull Request

Creas una solicitud de integración donde el equipo revisa tu código antes de aprobarlo.

Python R

Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

Python: El Lenguaje Versátil

Python se ha consolidado como el lenguaje preferido para ciencia de datos gracias a su sintaxis clara, su filosofía de legibilidad y su ecosistema extraordinariamente rico de bibliotecas especializadas.

Creado en 1991 por Guido van Rossum, Python es un lenguaje de propósito general que se destaca en ciencia de datos, desarrollo web, automatización, inteligencia artificial y muchas otras áreas.

Su comunidad activa y su vasta documentación lo hacen ideal tanto para principiantes como para expertos en el campo.

Facetas de Python

Comunidad

Machine Learning

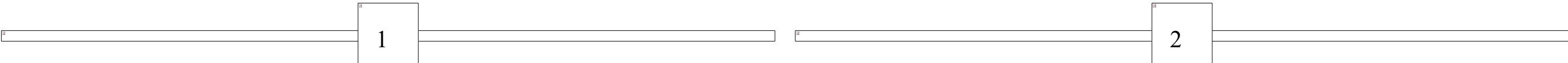
Big Data

Visualización

Python R

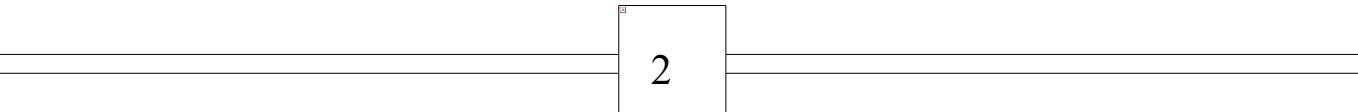
Comparación de lenguajes para ciencia de datos.

Bloque F: Herramientas para Proyectos de Ciencia de Datos



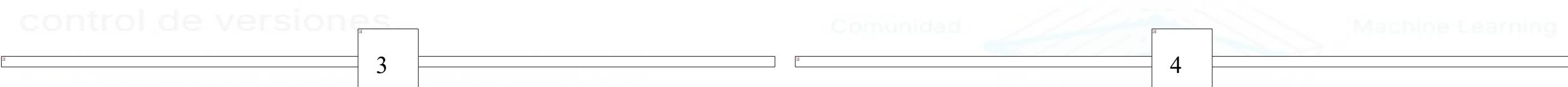
Sintaxis Legible

Código que parece pseudocódigo, reduciendo la curva de aprendizaje y facilitando el mantenimiento. La filosofía "Beautiful is better than ugly" hace que el código sea auto-documentado.



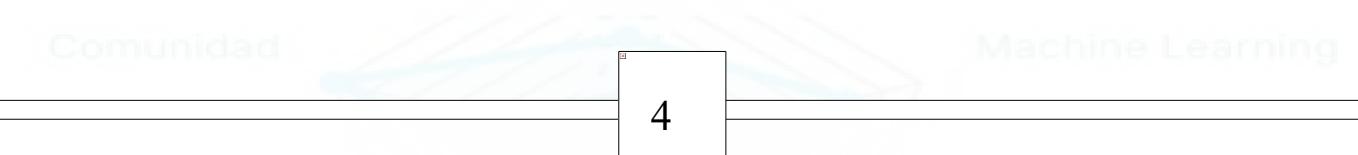
Ecosistema Rico

Miles de bibliotecas especializadas para cada tarea imaginable en ciencia de datos, desde manipulación de datos hasta aprendizaje profundo y visualización avanzada.



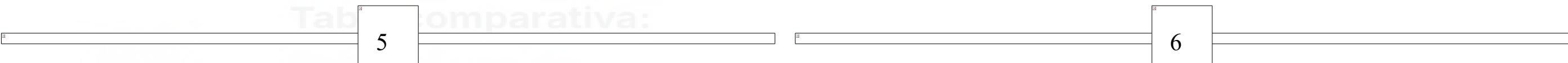
Comunidad Activa

Millones de desarrolladores en todo el mundo comparten conocimiento, resuelven dudas y contribuyen con nuevas herramientas constantemente.



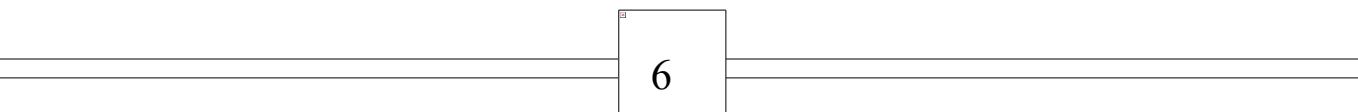
Versatilidad

No solo sirve para análisis de datos: puedes construir APIs, automatizar tareas, crear dashboards interactivos y desplegar modelos en producción.



Integración

Se conecta fácilmente con bases de datos, servicios web, otros lenguajes y herramientas empresariales, facilitando proyectos end-to-end.

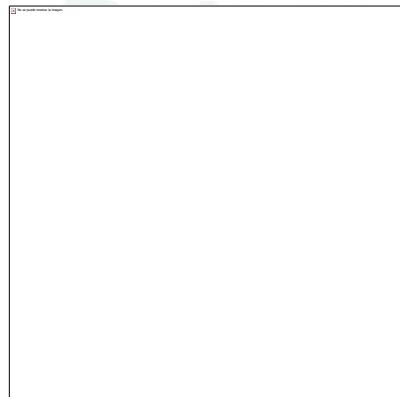


Gratis y Open Source

Sin costos de licencia y con código abierto que permite entender cómo funcionan las herramientas internamente.

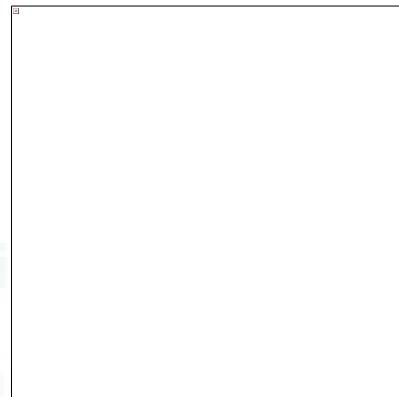
Comparación de lenguajes para ciencia de datos.

Bibliotecas Esenciales de Python



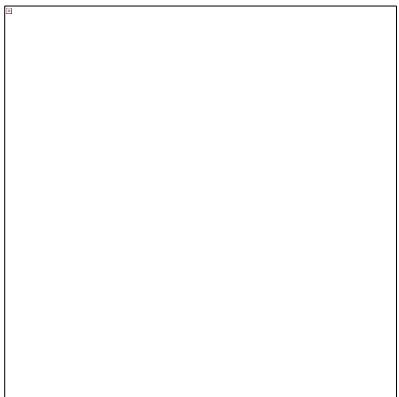
Pandas

La herramienta fundamental para manipulación y análisis de datos estructurados. Ofrece DataFrames potentes para limpiar, transformar y analizar datos de manera eficiente.



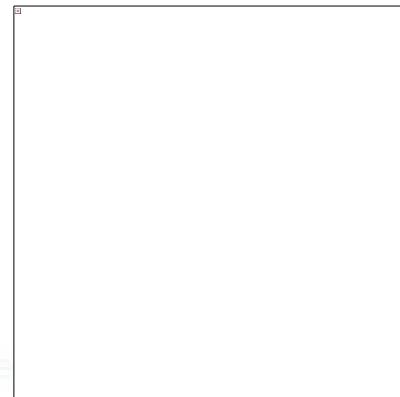
NumPy

La base de la computación numérica en Python. Proporciona arrays multidimensionales y operaciones matemáticas de alto rendimiento.



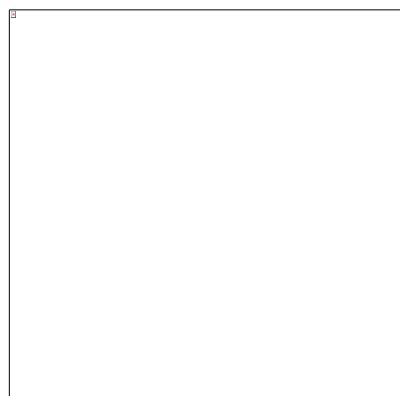
Matplotlib

Biblioteca de visualización versátil para crear gráficos estáticos, animados e interactivos de calidad publicable.



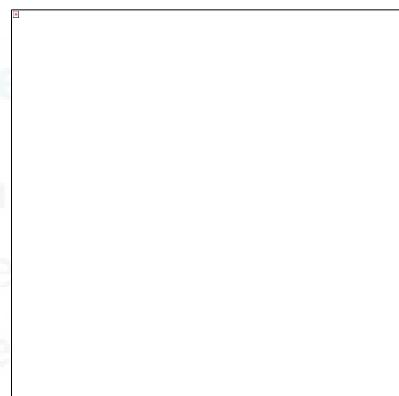
Scikit-learn

La biblioteca líder para aprendizaje automático, con implementaciones de algoritmos de clasificación, regresión, clustering y más.



Seaborn

Visualización estadística de alto nivel construida sobre Matplotlib, perfecta para exploración visual de datos complejos.



Jupyter

Entorno de notebooks interactivos que combina código, visualizaciones y documentación en un mismo documento.

Bloque F: Herramientas para Proyectos de Ciencia de Datos

R: El Lenguaje de los Estadísticos

R es un lenguaje y entorno de programación diseñado específicamente para computación estadística y visualización gráfica. Creado en 1993 por Ross Ihaka y Robert Gentleman, R se ha convertido en la herramienta preferida en ámbitos académicos, de investigación y análisis estadístico avanzado.

- **Plataformas para colaboración y**

Su sistema de paquetes CRAN (Comprehensive R Archive Network) contiene más de 19,000 paquetes especializados que cubren prácticamente cualquier método estadístico imaginable.

- **Ejemplos de aplicación del**

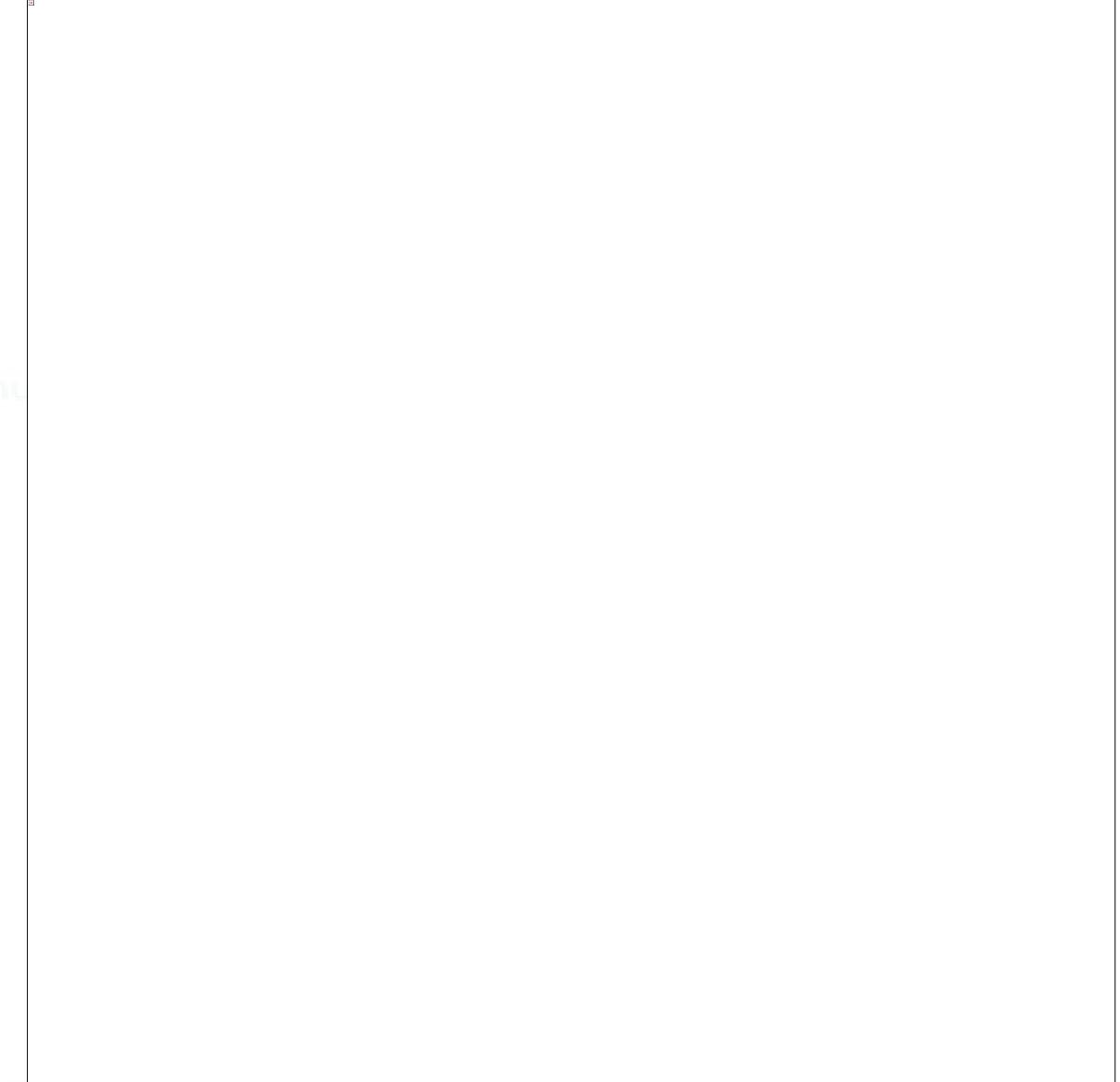
R brilla especialmente en análisis exploratorio de datos, modelado estadístico complejo y creación de visualizaciones de alta calidad para publicaciones científicas.

Tabla comparativa:

10%



Sobre el uso de diferentes lenguajes y arquitecturas, vinculada al Problema Prototípico.



Comparación de lenguajes para ciencia de datos.