

EL4106 - Inteligencia Computacional

Auxiliar 1

Profesor: Pablo Estévez V.

Auxiliar: Pablo Cornejo M.

Ayudantes: Diego Castillo, Andrés González, Sebastián Guzmán, Francisco Soto

Contenidos del Curso

1. Principios de Inteligencia Artificial

- Definición IA
- Fundamentos
- Tipos de Aprendizaje
- Técnicas
- Aplicaciones
- Limitaciones
- Ética

2. Clasificación

- Reconocimiento de Patrones
- Conjuntos de Datos
- Clasificación Bayesiana
- Clasificadores
- Funciones de Activación
- Algoritmos
- Funciones de Costo
- Entrenamiento
- Modelos
- Pre-procesamiento
- Medidas de Desempeño
- Generalización

Contenidos del Curso

3. Deep Learning

- Redes Neuronales Convolucionales
- Filtros y Pooling
- Arquitecturas
- Optimización
- Batch Normalization
- Autoencoders
- Redes Neuronales Recurrentes
- Aplicaciones

4. Clustering

- Cuantización Vectorial
- K-means
- Clustering Aglomerativo
- DBSCAN
- Mapas de Kohonen
- Validación
- PCA y Kernel PCA
- t-SNE y UMAP
- Aplicaciones

Clases Auxiliares

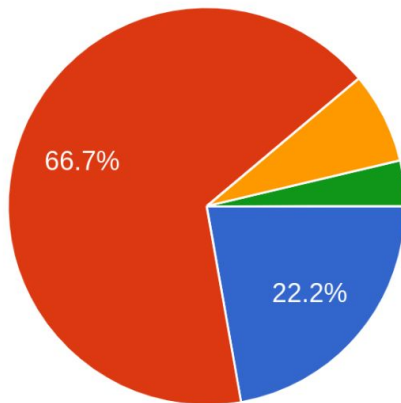
- Auxiliar 1 Introducción y Conceptos Fundamentales (16/08/2024)
- Auxiliar 2 Introducción a Pytorch (23/08/2024)
- Auxiliar 3 Aplicación a CNNs (29/08/2024)
- Presentación Proyectos (30/08/2024)

Información disponible en Calendario 2024 EI4106

Resultados Encuesta

¿Cuál es tu nivel de experiencia con el lenguaje de programación Python?

27 respuestas

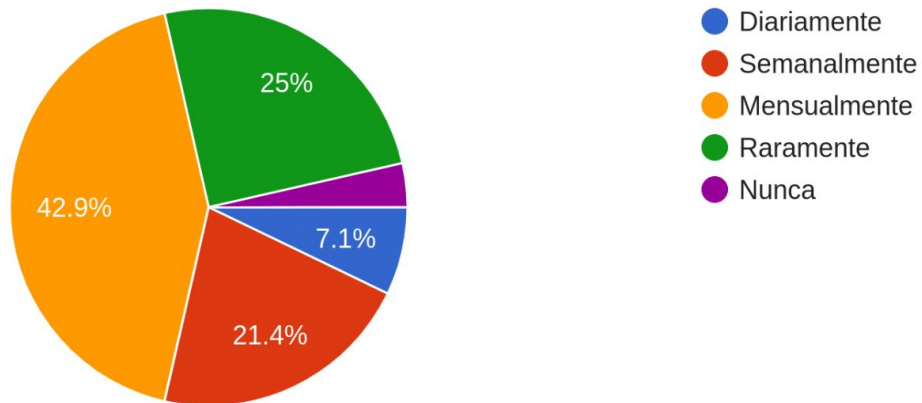


- Principiante: Conozco los conceptos básicos de Python y puedo escribir scripts simples
- Intermedio: Puedo usar Python para proyectos más complejos y tengo experiencia con bibliotecas comunes.
- Avanzado: Tengo experiencia en el desarrollo de aplicaciones complejas y en el uso de bibliotecas avanzadas.
- No tengo experiencia

Resultados Encuesta

¿Con qué frecuencia utilizas Python para tus tareas o proyectos?

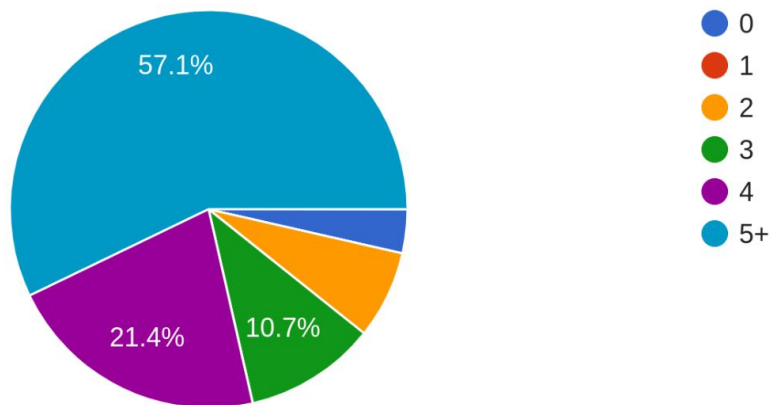
28 respuestas



Resultados Encuesta

¿En cuantos cursos has utilizado python para desarrollar tareas o proyectos?

28 respuestas



Repaso General

Características Clave:

- **Lenguaje de Programación Orientado a Objetos:** Python es un lenguaje de programación multiparadigma que soporta la programación orientada a objetos.
- **Versión Estable Actual:** La versión estable actual de Python es la **3.12.4**, lanzada el **6 de junio de 2024**. Se recomienda utilizar versiones 3.x y evitar versiones 2.x a menos que sea necesario.
- **Archivos de Script:** Los scripts de Python se guardan en archivos con extensión **.py** y se ejecutan utilizando el intérprete de Python.

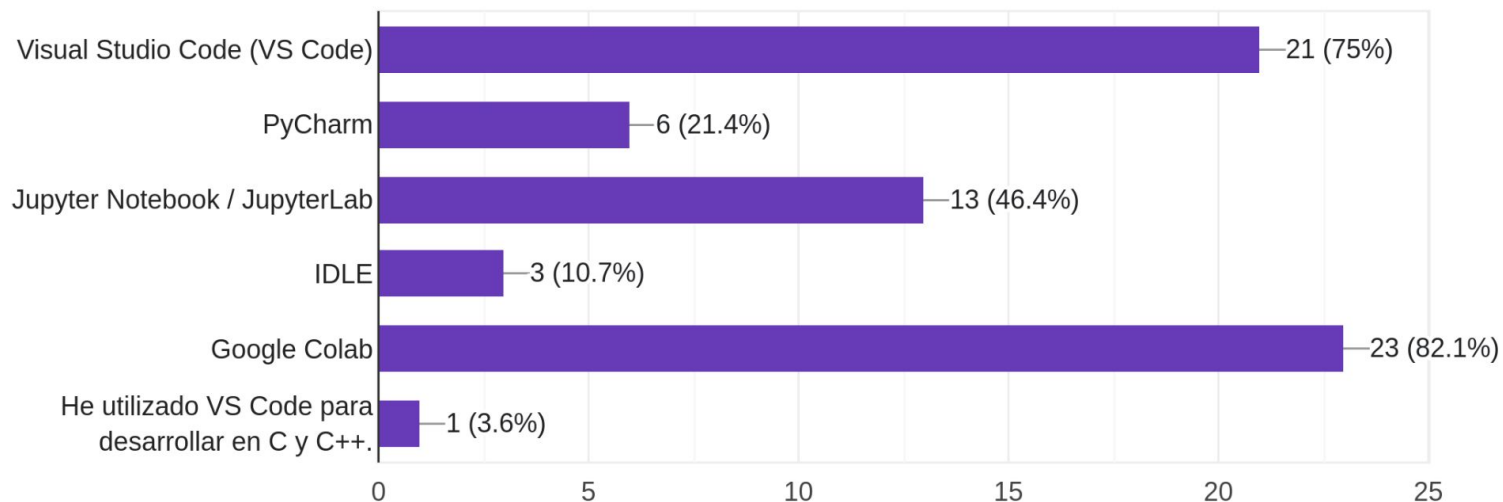
Links:

- Tutorial Python: <https://docs.python.org/es/3/tutorial/>

Resultados Encuesta

¿Qué entornos de desarrollo utilizas para desarrollar en Python? (Selecciona todas las opciones que apliquen)

28 respuestas



Editores de Código

Se recomienda utilizar editores de código con funcionalidades avanzadas, como:

- [Visual Studio Code](#) (VSCode): Editor ligero y versátil, con gran variedad de extensiones, ofrece integración con Git.
- [PyCharm](#): IDE con funcionalidades más avanzadas, ideal para proyectos de mayor escala, versión gratuita (Community).

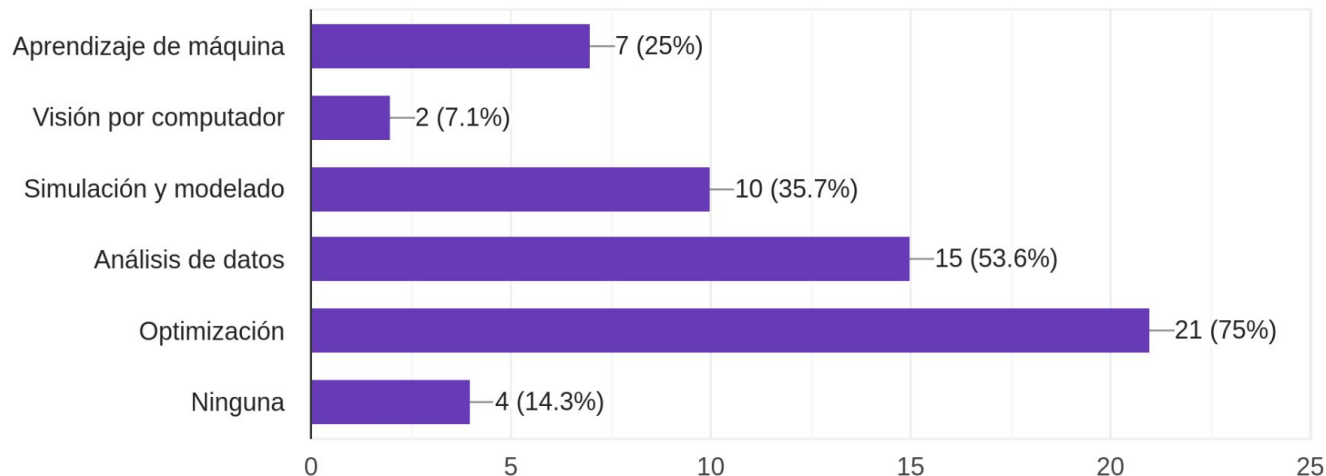
Estos editores ofrecen características como autocompletado, resaltado de sintaxis, refactorización, entre otras, que facilitan el desarrollo en Python.

- [Jupyter Notebook](#): Herramienta basada en web para crear y compartir documentos que contienen código, ecuaciones, visualizaciones y texto narrativo.

Resultados Encuesta

¿Qué tipo de tareas de Inteligencia Computacional has realizado utilizando Python? (Selecciona todas las que apliquen)

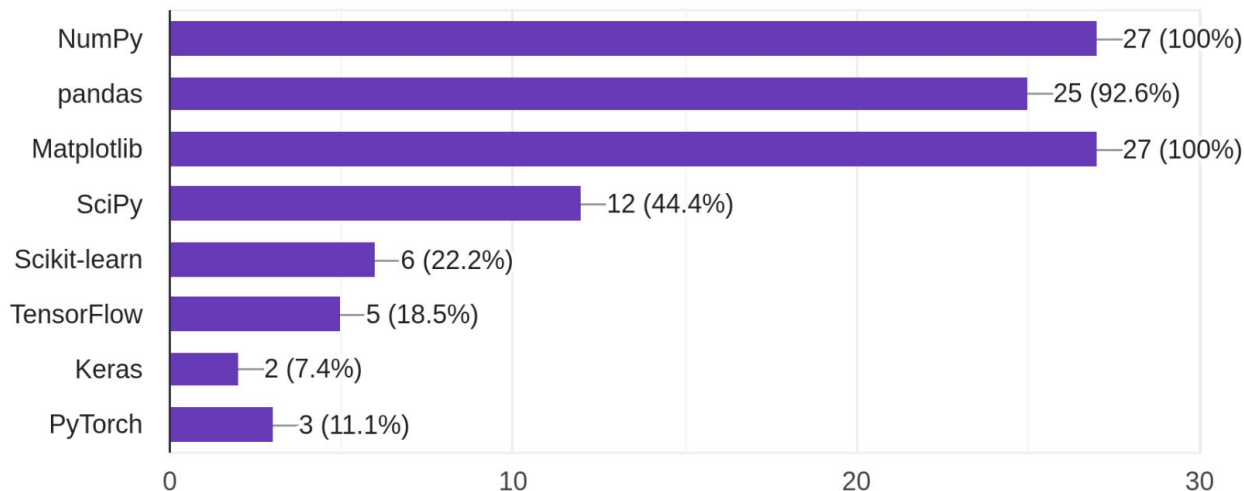
28 respuestas



Resultados Encuesta

¿Qué tan familiarizado estás con las siguientes bibliotecas y herramientas de Python?
(Selecciona todas las que conozcas)

27 respuestas



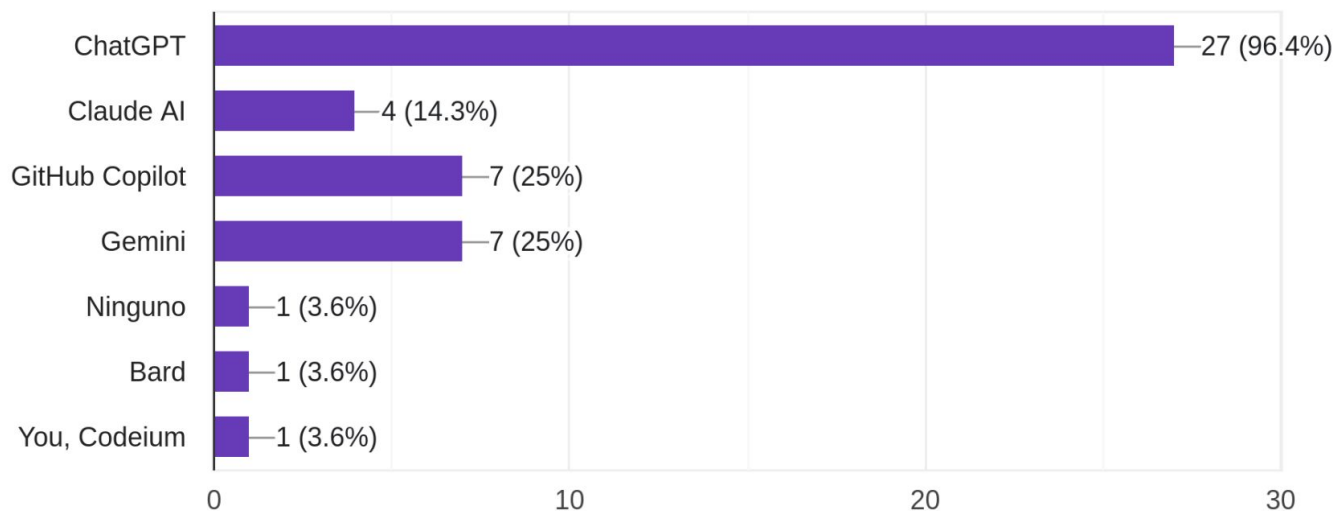
Evaluaciones Del Curso

- **Actividades:** Entregable de Códigos
 - Actividad 1: Python y librerías numéricas
 - Actividad 2: Pytorch
 - Actividad 3: Métricas de evaluación
 - Actividad 4 (Recuperativa): Revisión de Conceptos
- **Tareas:** Entregable de un informe (pdf) y códigos desarrollados en el trabajo.
 - Tarea 1: Perceptron Multicapa
 - Tarea 2: CNN
 - Tarea 3: LSTM, GRU
 - Tarea 4: SVM y Random Forest
 - Tarea 5 (Recuperativa): Aprendizaje no supervisado
- **Proyecto:** A conversar en Presentación Proyectos (30/08/2024)

Resultados Encuesta

¿Qué herramientas de asistencia en programación basadas en inteligencia artificial utilizas?

28 respuestas



Herramientas de IA para el Desarrollo de Código

Las herramientas de IA pueden mejorar la productividad y eficiencia de los programadores,

- GitHub Copilot: Asistente de IA entrenado por Anthropic que sugiere código y completar líneas a medida que el programador escribe.
- ChatGPT (OpenAI): ChatGPT es un modelo de lenguaje desarrollado por OpenAI que utiliza la arquitectura GPT-4 para generar texto a partir de instrucciones en lenguaje natural. Aunque su principal uso es la generación de texto, también puede ser útil para el desarrollo de código, proporcionando sugerencias, explicaciones y resolviendo dudas sobre programación.
- Claude (Anthropic): Claude es un asistente de IA creado por Anthropic, diseñado para ser seguro y alineado con los valores humanos. Claude es capaz de asistir en tareas de generación de texto y puede ser aplicado a la escritura de código, similar a ChatGPT.

¿Cómo correr scripts?

- Los scripts de python se ejecutan utilizando el intérprete de python. Este intérprete se instala cuando uno instala python y puede ser ejecutado escribiendo:

```
python mi_archivo.py
```

- Desde la terminal en Linux y MAC OS, y desde la línea de comando (cmd) en Windows.
- Algunos editores de texto también permiten ejecutar scripts de forma automática, pero es importante saber que por detrás esto es lo que está sucediendo siempre que se ejecuta un script de python.

Organización de Código

- Cuando uno corre un script de python se crea lo que se llama el python path. Esto es una lista de todos los directorios donde python puede encontrar archivos para importar.
- Por defecto, siempre se incluye dentro del python path al directorio desde donde se está corriendo el script, es por esto que si hay otros scripts a importar, deben estar dentro del mismo directorio.
- Si se quiere reutilizar un código como si fuese un módulo en cualquier parte, es necesario convertirlo en un package que pueda ser instalado.
- Si en algún momento tienen un error de importación de un módulo, intentencorriendo el script desde la carpeta donde se encuentra este y los demás a importar.

Instalación de Paquetes

- El instalador de paquetes por defecto de python es pip (Package Installer for Python). Este permite encontrar paquetes en el repositorio PyPi (Python Package Index) e instalarlos.
- También existe Anaconda, el cual es estrictamente una distribución de Python (Python se instala desde Anaconda si optan por esta ruta). Anaconda está pensado principalmente para data science y el instalador de paquetes que se utiliza por defecto es conda (también puede utilizarse pip pero no se recomienda).
- Para instalar anaconda pueden seguir el siguiente [link](#).

Instalación de Paquetes (conda)

- Para crear un ambiente virtual con conda, deben correr la siguiente línea desde la terminal:

```
conda create -n envname python=x.x
```

- Para activar y desactivar un ambiente correr los siguientes comandos (respectivamente):

```
conda activate envname
```

```
conda deactivate
```

- Para instalar librerías, correr el siguiente comando:

```
conda install numpy
```

- Para agregar un ambiente a jupyter (para ser usado

```
ipython kernel install --user --name=envname
```

Instalación de Paquetes (pip + venv)

- Para crear un ambiente virtual con venv, deben correr la siguiente línea desde la terminal:

```
python -m venv envname
```

- Para activar y desactivar un ambiente correr los siguientes comandos (respectivamente):

```
windows: envname\Scripts\activate
```

```
linux y macos: source ./envname/bin/activate
```

```
deactivate
```

- Para instalar librerías, correr el siguiente comando:

```
python -m pip install numpy
```

- Para agregar un ambiente a jupyter (para ser usado del navegador):

```
python kernel install --user --name=envname
```

Buenas y Malas Prácticas

- Es recomendable separar los archivos .py en archivos que serán corridos directamente y archivos que definen clases y funciones a ser importadas.

De todas maneras, si crean un archivo que luego correrán, se recomienda incluir el código de la siguiente manera:

```
1
2 def main():
3     # Código a correr
4     pass
5
6
7 if __name__ == "__main__":
8     main()
```

- Este if del final sólo se ejecutará cuando se corra el archivo de forma directa. Si ustedes incluyen el código a correr sin este if, entonces el código se ejecutará incluso si ustedes importan este archivo en otro script.