SANJANA DEVI

AI/ML Student | Intern @ CodroidHub Private Limited | First-Year Engineering Student.

# Artificial Intelligence/Machine Learning Bootcamp

From Fundamentals to Real-World AI — Your Machine Learning Journey Starts Here.

## Agenda

Welcome to your Python and Jupyter learning journey. This notebook will walk you through essential concepts to set up, understand, and maximize your development environment.

---

### Topics Covered

---

> Use the links above to jump directly to any section in this notebook.

Set the Stage for Success — Your Python Environment Starts Here.

## Environment SetUp

To use Jupyter Notebook effectively, you need to have Python or Anaconda installed on your system. Below are the step-by-step instructions for setting up the environment across different operating systems.

---

### Option 1: Installing Python + pip (Recommended for developers)

#### Step 1: Download & Install Python

- Visit the official website: https://www.python.org/downloads/
- Choose your OS: Windows, macOS, or Linux
- ✅ **Important**: During installation, **check the box**:
  **“Add Python to PATH”**

#### ⚙ Step 2: Verify Python Installation

Open your terminal or command prompt and run:

```
python --version
pip --version
```

### ⚙ Step 3: Install Jupyter Notebook

```
pip install notebook
```

### ⚙ Step 4: Launch Jupyter Notebook

```
jupyter notebook

This opens in your browser at: http://localhost:8888

Use the interface to create or open .ipynb files
```

**OR — Alternative Option Below**

You may follow the next method as an alternate setup approach.

## Option 2: Anaconda Installation Guide (Windows / macOS / Linux)

Anaconda is a popular Python distribution that comes bundled with Jupyter Notebook, conda, and essential data science libraries. It's highly recommended for beginners and data professionals.

---

### Step 1: Download Anaconda

- Visit the official website: https://www.anaconda.com/products/distribution
- Click **Download** under the **Individual Edition**
- Choose the installer for your operating system:
  - Windows (.exe)
  - macOS (.pkg)
  - Linux (.sh)

---

### Step 2: Install Anaconda

#### For Windows:

- Run the `.exe` installer
- ✅ Check: **“Add Anaconda to my PATH environment variable”** (recommended)
- Proceed with the installation (default settings are fine)

#### For macOS:

- Run the `.pkg` installer
- Follow the instructions
- Restart your terminal after installation

#### For Linux:

- Open terminal in the download directory
- Run:

```
bash Anaconda3-<version>-Linux-x86_64.sh
```

Where Code Meets Clarity — Welcome to the World of Jupyter Notebooks.

## Introduction to Jupyter Notebook

**Jupyter Notebook** is a powerful, open-source interactive development environment (IDE) primarily used for **data analysis**, **scientific computing**, **machine learning**, **exploratory programming**, and **education**.

It supports mixing **code**, **visualizations**, and **narrative text (Markdown)** in a single, shareable document, making it ideal for both prototyping and presentation.

---

## Key Features

- **Interactive Code Execution**: Run code one cell at a time and see output immediately.
- **Language Support**: Though widely used for Python, it supports over 40 programming languages via *Jupyter kernels* (e.g., R, Julia, Bash).
- **Markdown & LaTeX Support**: Ideal for writing rich documentation and mathematical formulas.
- **Data Visualization**: Integrates seamlessly with libraries like `matplotlib`, `seaborn`, `plotly`, and `bokeh`.
- **Notebook Sharing**: Save and share `.ipynb` files or export as PDF/HTML slides.
- **Extensibility**: Supports plugins like **nbextensions**, **widgets**, **voilà**, and more.

---

## Typical Use Cases

- Data cleaning and transformation
- Numerical simulation and modeling
- Visualization and storytelling
- Machine learning model training and evaluation
- Teaching and interactive assignments

---

## How Jupyter Notebook Compares with Other Environments

| Feature | Jupyter Notebook | Google Colab | VS Code (Python Extension) |
|---|---|---|---|
| **Setup Required** | Yes | No (cloud-based) | Yes |
| **Runs in Browser** | Yes (locally) | Yes (cloud) | No (runs locally with GUI) |
| **Free GPU Access** | No | ✅ Yes (limited quota) | ❌ No |
| **Offline Usage** | ✅ Yes | ❌ No | ✅ Yes |
| **Integrated Terminal & Git** | Limited | No | ✅ Full terminal, Git support |
| **Extensibility** | High (nbextensions, widgets, etc.) | Medium | High (extensions, LSPs, linters) |
| **Markdown & LaTeX Support** | ✅ Yes | ✅ Yes | ✅ (with Markdown Preview plugin) |
| **Deployment Ready** | ❌ No | ❌ No | ✅ Yes (production-ready development) |

---

## Summary

Jupyter Notebook excels in **interactive computing**, **data analysis**, and **rapid experimentation**. It's favored by data scientists, educators, and researchers for its ability to combine computation with documentation in one coherent file.

While tools like **VS Code** are better suited for **full-stack development and large-scale projects**, and **Google Colab** offers convenient **cloud-based execution with free GPU**, Jupyter remains a go-to tool for **local prototyping, research notebooks, and presentations**.

---

> For best experience, combine Jupyter with tools like `nbextensions`, `nbconvert`, `Voila`, and GitHub for version control and sharing.

## Master the Keys to Speed — Essential Jupyter Notebook Shortcuts at Your Fingertips

# Essential Jupyter Notebook Shortcuts

## Ensure you're in **Command Mode** (press `Esc`) before using these.

---

## Add New Cells

| Action | Shortcut |
|---|---|
| Add cell below | `B` |
| Add cell above | `A` |

---

## Switch Cell Type

| Action | Shortcut |
|---|---|
| Convert to Markdown | `M` |
| Convert to Code | `Y` |

---

## Run & Move

| Action | Shortcut |
|---|---|
| Run selected cell | `Shift + Enter` |
| Run and insert cell below | `Alt + Enter` |
| Run and stay in same cell | `Ctrl + Enter` |

---

## Other Handy Shortcuts

| Action | Shortcut |
|---|---|
| Delete cell | `D` then `D` |
| Copy cell | `C` |
| Paste cell | `V` |
| Cut cell | `X` |
| Undo deleted cell | `Z` |
| Move cell up | `K` then `Shift+K` |
| Move cell down | `J` then `Shift+J` |

---

## View All Shortcuts

> To see a full list of shortcuts:
>
> - Press `H` in **Command Mode** ( `Esc` to exit editing first)

Python Versions: Evolution of Simplicity, Power, and Performance.

# Python Versions

## Python Versions: Overview & Key Comparisons

Python is a high-level, general-purpose programming language known for its readability, flexibility, and vast ecosystem. Over time, Python has evolved significantly through version upgrades.

---

## Major Python Versions

### Python 2.x (Legacy, EOL in 2020)

- Released: 2000
- Syntax was less consistent than Python 3
- **No longer supported** as of January 1, 2020
- Popular tools like TensorFlow and Pandas dropped support

### Python 3.x (Current & Actively Maintained)

- Introduced: 2008
- Cleaner syntax, better Unicode support, and new libraries

- Recommended for all new development

---

## Key Differences: Python 2 vs Python 3

| Feature | Python 2 | Python 3 |
|---|---|---|
| `print` Statement | `print "Hello"` | `print("Hello")` |
| Integer Division | `3/2 = 1` | `3/2 = 1.5` |
| Unicode Support | Limited | Native `str` is Unicode |
| `xrange()` | Exists | Replaced with `range()` |
| Community Support | Ended in 2020 | Actively supported |
| Package Compatibility | Decreasing | Increasing and modernized |

---

## Python 3.x Minor Versions

| Version | Highlights | Status | |-----------|---------------------------------------------------|--------------| | 3.6 | f-strings, type hints improvements | Ended | | 3.7 | Data classes, built-in `breakpoint()` | Ended | | 3.8 | Walrus operator ( `:=` ), positional-only args | Ended | | 3.9 | Dictionary merge ( `|` ), type hinting updates | Ended | | 3.10 | Structural Pattern Matching ( `match-case` ) | Active | | 3.11 | Performance boosts, error trace improvements | Active | | 3.12 | Runtime auditing, more syntax optimization | Latest (2023+)|

---

## ⚠ Best Practices

- Always use the **latest stable version** (currently 3.12+) for new projects.
- For compatibility with ML libraries like TensorFlow, use 3.8–3.11.
- Check your version using:

```
python --version
```

Behind Every Smart System, There's Python — The Engine of Modern Technology.

# Python Use Cases & Industry Applications

Python is one of the most versatile and widely adopted programming languages in the world. It powers applications across industries — from startups to tech giants — due to its simplicity, readability, and an extensive ecosystem of libraries.

---

## ✅ Common Use Cases of Python

| Domain | Applications | Popular Libraries |
|---|---|---|
| **Data Science** | Data analysis, visualization, forecasting | `pandas`, `matplotlib`, `seaborn`, `numpy` |
| **Machine Learning / AI** | Model training, natural language processing, deep learning | `scikit-learn`, `TensorFlow`, `PyTorch`, `transformers` |
| **Web Development** | Server-side logic, REST APIs, CMS, dashboards | `Flask`, `Django`, `FastAPI` |
| **Automation / Scripting** | Task automation, system operations, data scraping | `os`, `shutil`, `selenium`, `beautifulsoup4` |
| **DevOps & SRE** | Deployment scripts, monitoring tools, CI/CD pipelines | `fabric`, `Ansible`, `invoke` |
| **Cybersecurity** | Pen testing, log analysis, threat detection | `scapy`, `shodan`, `requests`, `socket` |
| **Game Development** | Rapid prototyping, backend game logic | `pygame`, `panda3d` |
| **Finance & FinTech** | Algorithmic trading, risk analysis, fraud detection | `pandas`, `quantlib`, `statsmodels` |
| **IoT & Robotics** | Sensor data processing, control systems | `RPi.GPIO`, `OpenCV`, `pySerial` |

---

## Companies Using Python & Their Use Cases

| Company | Use Case / Purpose | Python Role |
|---|---|---|
| **Google** | Backend services, AI/ML research, internal tools | TensorFlow (developed by Google) |

| | | |
|---|---|---|
| [illegible] | [illegible] | [illegible] |
| **Netflix** | Recommendation systems, data analytics, automation | Data pipelines, ML models |
| **Spotify** | Music recommendation, backend APIs | Data science, web backend |
| **Facebook (Meta)** | Infrastructure automation, AI research | PyTorch (developed by Facebook) |
| **Instagram** | Web backend (originally built on Django) | API server, content pipelines |
| **Dropbox** | Desktop clients and cloud sync logic | Cross-platform app development |
| **NASA** | Scientific calculations, data analysis | Mission analysis and data workflows |
| **Reddit** | Community platform backend | Web development (Python + Flask) |
| **Quora** | Content ranking, web backend | Backend logic with performance tuning |
| **Uber** | Forecasting demand, pricing algorithms | ML models, internal tools |
| **Industrial Light & Magic** | Visual effects rendering | Image processing and automation |

---

## Why Companies Choose Python

- **Fast prototyping** and development
- **Huge library ecosystem** for any domain
- **Community support** and open-source
- **Readable syntax** for collaboration and maintainability
- Integrates well with **C/C++**, **Java**, **cloud services**, and **databases**

---

> ⚙ Whether you're building AI models, automating workflows, or scaling backend services — Python is a tool that adapts to your needs.

---

Python Unlocks Potential — No Matter Your Field, There’s a Script for It.

# Job Scope of Python & Cross-Domain Opportunities

Python is more than just a programming language — it's a gateway to thriving tech and non-tech careers across industries. Its simplicity and power make it a perfect tool for professionals from **any background**, including **biotechnology**, **mechanical engineering**, **finance**, and more.

---

## Why Python Has Massive Career Scope

- Used across industries: tech, healthcare, energy, education, finance, and more
- Demand for Python developers continues to rise due to AI, ML, and automation
- □ Easy to learn, powerful to build — ideal for both beginners and professionals
- Open-source and community-supported with vast library ecosystem

---

## Popular Career Roles with Python

| Role | Where Python Fits |
|---|---|
| **Data Scientist / Analyst** | Data cleaning, analysis, machine learning |
| **AI/ML Engineer** | Model development and deployment |
| **Web Developer** | Backend logic and REST APIs |
| **DevOps Engineer** | Automation scripts, CI/CD pipelines |
| **Cybersecurity Analyst** | Log analysis, threat detection, pen testing |
| **Bioinformatician** | DNA sequence analysis, data visualization |
| **Mechanical Simulation Engineer** | Automation, simulation control, visualization |
| **Financial Analyst** | Risk modeling, algorithmic trading |
| **Educator / Researcher** | Teaching, simulations, data processing |
| **IoT Developer** | Device programming and data transmission |

---

## Python for Other Domain Experts

### Biotechnology & Bioinformatics

- Analyzing gene expression data using `Biopython`, `pandas`, `scikit-learn`
- Simulating protein interactions and biological models
- Drug discovery pipelines and visualizations

### ☐ Mechanical & Civil Engineering

- Automating CAD model tasks using Python APIs
- Simulation & control using Python with tools like `Matplotlib`, `SimPy`, `PyBullet`
- Data logging from sensors and real-time visualization

### Finance & Economics

- Building financial dashboards and trade bots
- Risk assessment and predictive modeling
- Using libraries like `QuantLib`, `pandas-datareader`, `yfinance`

### Education & Research

- Creating interactive simulations
- Statistical computing and automation of grading tools
- Building educational apps with `Tkinter`, `Streamlit`

### ⚙ Manufacturing & Robotics

- Automation scripts for industrial processes
- Robotics using `ROS`, `OpenCV`, and Python-based microcontrollers
- Integrating sensors with Python (e.g., Raspberry Pi + Python)

---

## Final Thoughts

Python acts as a **bridge** between domain expertise and modern computing:

> “Domain knowledge + Python = Career multiplier”

You don’t need to become a full-stack developer to use Python effectively. You just need to apply Python to **your own field** to solve real problems faster and smarter.

---

In [ ]:
```
# Summary Summary
```

**Connect @**
Mail (Sanjana): sanjanadevibihana@gmail.com
Contact: +91-6283762268

SANJANA DEVI

AI/ML Student | Intern @ CodroidHub Private Limited | First-Year Engineering Student.

SANJANA DEVI

In [ ]:
```
```