Python for AI Beg inners

Code Examples and Fundamentals

Amina Mujawar

CONTENT

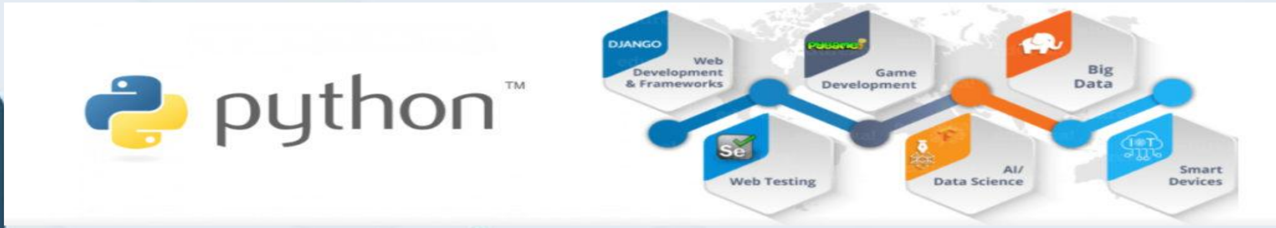
What Is Python

Python is an open-sourced, interpreted, object-oriented, high-level programming language with dynamic syntax.

It is highly attractive for Rapid Application Development and scripting.

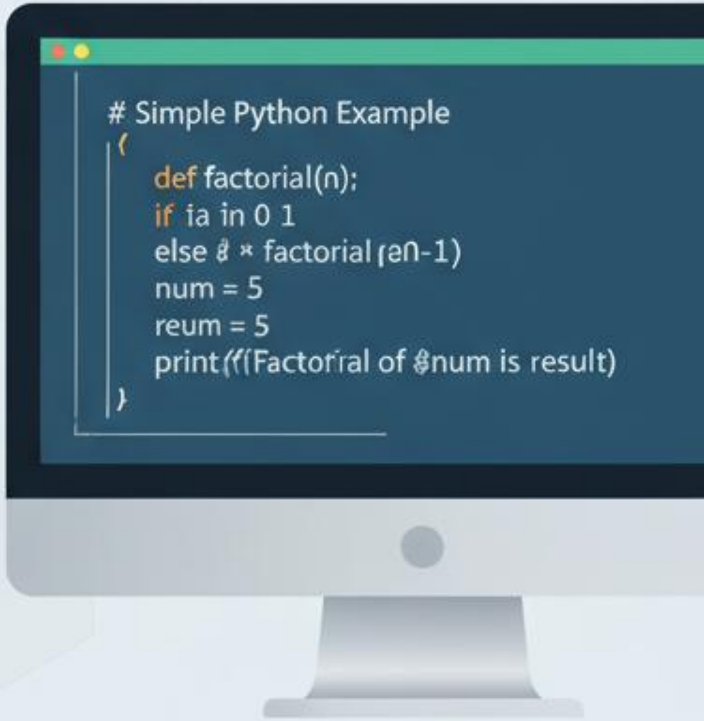
It was initially formulated by **Guido van Rossum** in the late **1980s** at Centrum Wiskunde & Informatica(CWI) in Netherland as a successor to the ABC language. The name Python was named after a BBC's TV Show called '**Monty Python's Flying Circus**' which he was a fan of.

https://www.youtube.com/watch?v=qxMcGDnT8uc



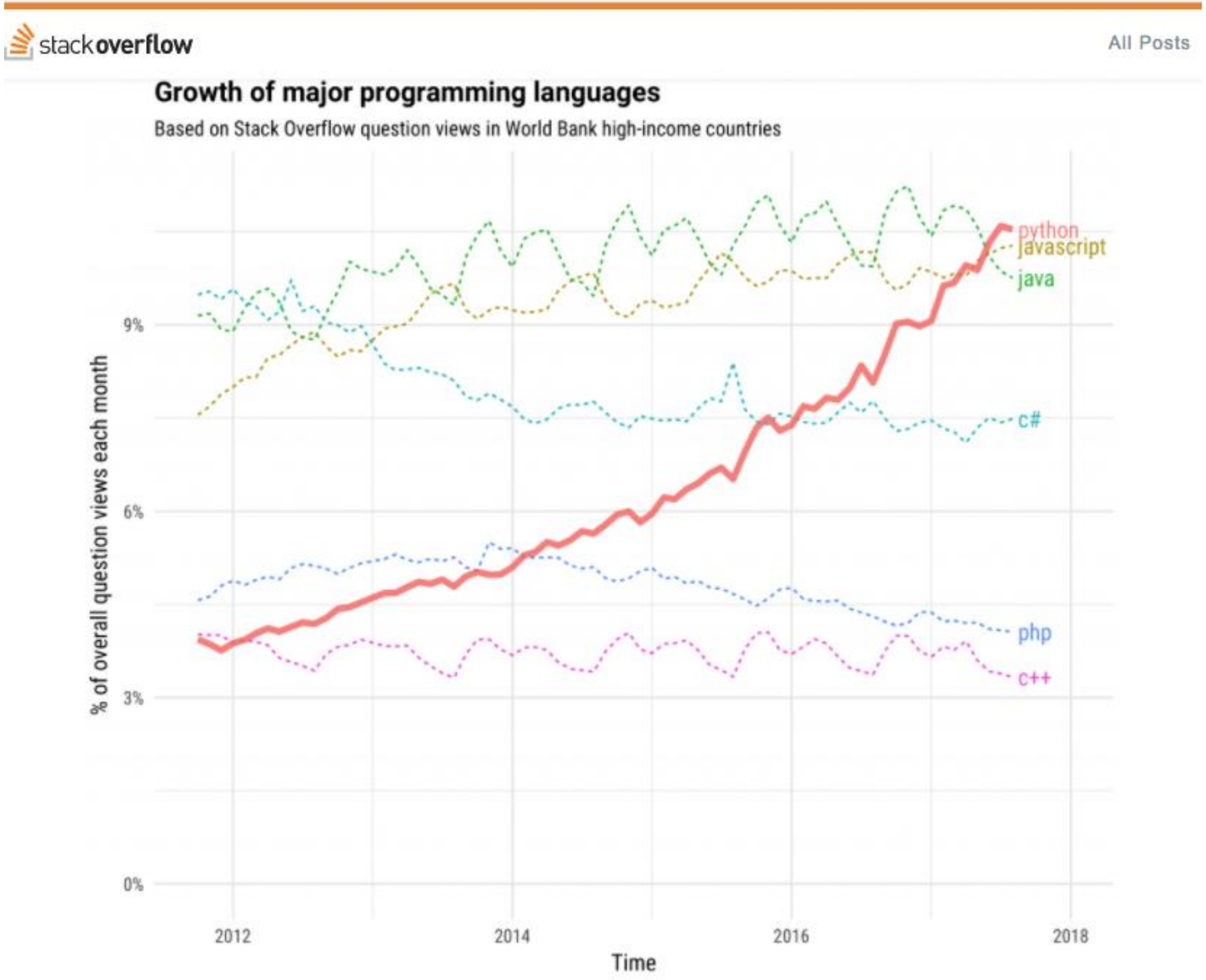
Easy to learn & use

Readable,
Simple,



```
# Simple Python Example
(
    def factorial(n):
    if  ia in 0 1
    else ꬾ × factorial (en-1)
    num = 5
    reum = 5
    print(f(Factorial of $num is result)
}
```

Increases productivity and reduces the cost of maintenance.


stackoverflow
All Posts
Growth of major programming languages
Based on Stack Overflow question views in World Bank high-income countries
% of overall question views each month
9%
6%
3%
0%
2012
2014
2016
2018
Time
python
javascript
java
c#
php
c++

Domain	Description
Desktop and Web Applications	A Desktop application is one that runs stand-alone in a desktop or laptop computer for example BitTorrent, Blender, Juice while a Web application requires a Web browser to run, for example Mailman, Phone, MoinMoin.
Data Science	It is a field that uses scientific methods such as data collection; algorithms and machine learning techniques to extract, analyze and process insights from raw data.
Machine Learning	It is an application of artificial intelligence (AI) that gives systems the ability to automatically learn and improve from experience and data without being explicitly programmed
Robotics	It is a branch of engineering that deals with the conception, design, manufacture, and operation of robots.
Artificial Intelligence	It is a broad field that deals with enabling machines to demonstrate intelligence similar to human's intelligence such as decision-making, facial recognition, etc. Artificial intelligence incorporates other fields like Machine Learning, Robotics, Natural Language Processing(NLP), etc.
Internet of Things (IoT)	It is a field that describes the network of things that are embedded with software, and other technologies for the purpose of connecting and exchanging data with other devices over the internet.
Gaming	It is the art of designing and programming games for entertainment, educational, or experimental purposes and that runs on computers and mobile devices.
Mobile Applications	It is a computer program or app designed to run on a mobile device such as a phone, table, or watch.
Natural Language processing	It is a field that analyses speech in both audible speech, as well as text of a language.

Some of the top features of Python include:

- Free and Open-Sourced
- Dynamically typed
- Portable
- Numerous libraries and applications
- Large supportive community
- Flexibility
- Easy to use and learn
- Extensible
- Embeddable
- Shorter line of code than most languages

Some Drawbacks of Python are:

•Slow speed

•Memory inefficient

•Ineffective in mobile computing.

•Undeveloped database layers.

•Run time error prompt due to its dynamism.

Understanding IDEs and Code Editors

Integrated Development Environments (IDEs) are specialized software applications designed for software development. They combine various tools to improve coding efficiency and productivity. A well-structured coding environment allows developers to streamline their workflow, enhance code management, and ultimately boost their productivity in software development.

Understanding Integrated Development Environments (IDEs)

What is an IDE?

An IDE provides a comprehensive platform for software development.

Key Features

Includes code editors with auto-completion and syntax highlighting.

Debugging Tools

Integrates debugging tools to identify and fix code errors efficiently.

Essential Features of IDEs

Code Saving & Reloading

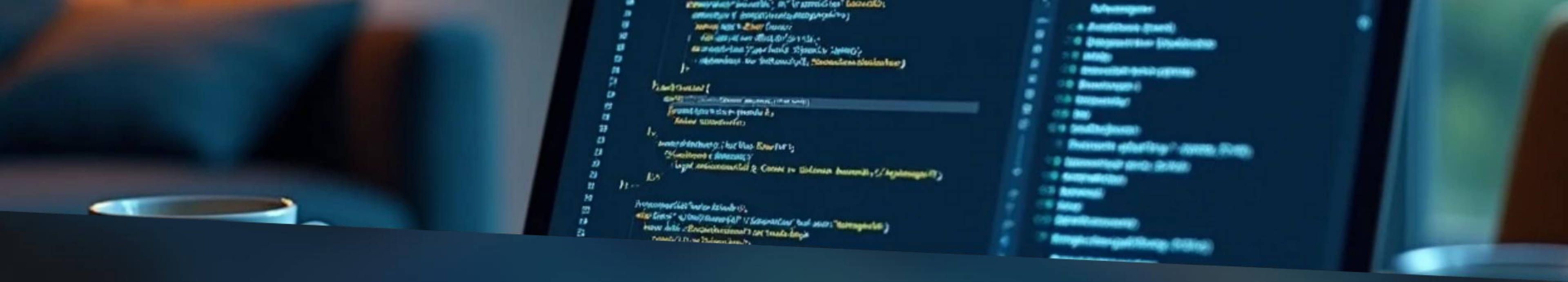
Saving and reloading code files is crucial for any IDE. Developers need the ability to store their work and resume later without losing any progress.

Debugging Support

Debugging support is fundamental, allowing developers to step through their code and identify issues in real-time, ensuring that their software runs as intended.

Syntax Highlighting

Syntax highlighting helps developers quickly identify keywords, variables, and symbols, improving code readability and comprehension.



Python Coding Environment Needs

Code Formatting Support

Automatically formats code, suggesting indentation and recognizing structures.

Version Control Integration

Built-in tools for managing code changes and tracking project history.

Integrated Terminal Functionality

Allows running scripts and commands directly from the IDE.

Popular Python IDEs

This presentation highlights some popular Integrated Development Environments (IDEs) for Python programming, focusing on their unique features and benefits for developers.

1

PyCharm

Feature-rich IDE favored by developers.

2

Visual Studio Code

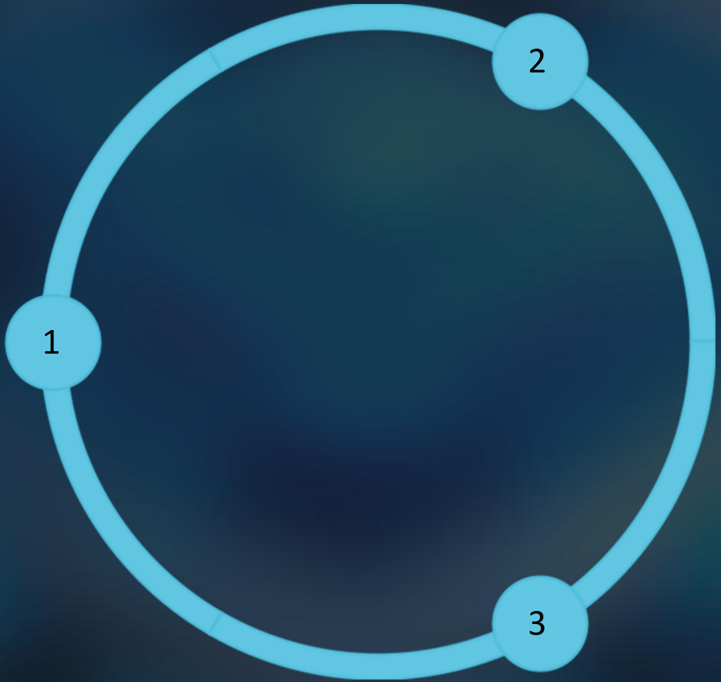
Lightweight and customizable with vast extensions.

3

Jupyter Notebook

Ideal for interactive coding in data science.

Choosing the Right IDE



Identify Project Needs

Consider the specific needs of your projects when choosing an IDE.

User-Friendly Design

Look for user-friendly interfaces and support for common Python libraries.

Community Support

Community support and documentation are essential in choosing an IDE.

Future of Coding Environments

1 Intelligent Features

The implementation of predictive coding and enhanced debugging tools will reduce developer workload.

2 Web-Based IDEs

Code from anywhere without complex setup procedures, making programming more accessible.

3 Collaboration Tools

Essential for remote work, enabling seamless teamwork on projects regardless of physical locations.

Conclusion on IDEs and Code Editors

Role in Development

Streamlining workflows and enhancing productivity.

Impact on Coding

Efficient coding and improved software quality.

Future Developments

Integration of new features for evolving developer needs.

01

Python Fundamentals for AI

Syntax and Basic Operations

Python Syntax Basics

Python uses indentation for code blocks instead of braces. Statements end with newlines rather than semicolons, making code clean and readable.

Variables and Data Types

Python variables require no declaration. Key data types include integers, floats, strings, booleans, lists, tuples, and dictionaries for AI applications.

Operators in Python

Python supports arithmetic (+,-,*,/), comparison (==,!=,<,>), logical (and,or,not), and assignment operators essential for AI algorithm implementation.

Control Flow Statements

If-else conditions and loops (for, while) control program execution flow. These structures are fundamental for implementing AI decision-making processes.

Data Types and Structures

Numeric Data Types

Python supports integers, floats, and complex numbers. Example: `x = 5`, `y = 3.14`, `z = 2+3j`. Essential for mathematical operations in AI algorithms.

String Manipulation

Text processing using strings. Example: `text = "AI Model"`. Strings support slicing, concatenation and methods like `split()` for natural language processing tasks.

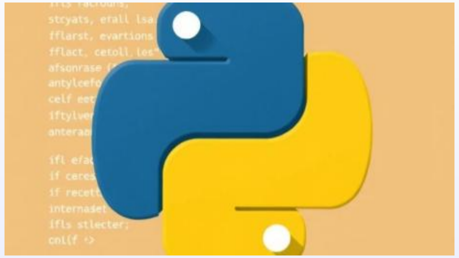
Lists and Arrays

Ordered, mutable collections. Example: `data = [1,2,3]`. NumPy arrays (`import numpy as np; arr = np.array([1,2,3])`) optimize mathematical operations for AI.

Boolean Logic

Boolean values (True/False) for conditional operations. Example: `is_trained = True`. Fundamental for decision-making in AI classification and filtering tasks.

Control Flow Mechanisms



Loops in Python

For and while loops enable repeated execution of code blocks, crucial for iterative processes in machine learning applications.

Try-Except Blocks

Error handling mechanism that prevents program crashes during AI model training and data processing operations.

01

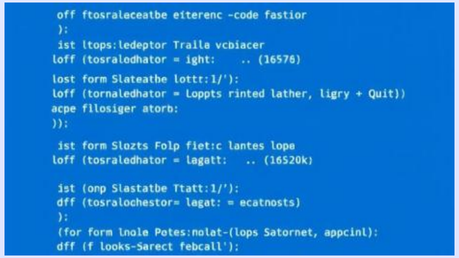
02

03

04

Conditional Statements

Python uses if, elif, and else statements to execute code based on conditions. Essential for decision-making in AI algorithms.



```
 off ftosraleceatbe eiterenc -code fastior
 ):
 ist ltops:ledeptor Tralla vcbiacer
loff (tosralodhator = ight:    .. (16576)
lost form Slateatbe lottt:1/'):
loff (tornaledhator = Loppts rinted lather, ligry + Quit))
acpe fllosiger atorb:
)):
 ist form Slozts Folp fiet:c lantes lope
loff (tosraledhator = lagatt:   .. (16520k)
 ist (onp Slastatbe Ttatt:1/'):
 dff (tosralochestor= lagat: = ecatnosts)
 ):
 (for form lnole Potes:nolat-(lops Satornet, appcinl):
 dff (f looks-Sarect febcall'):
```

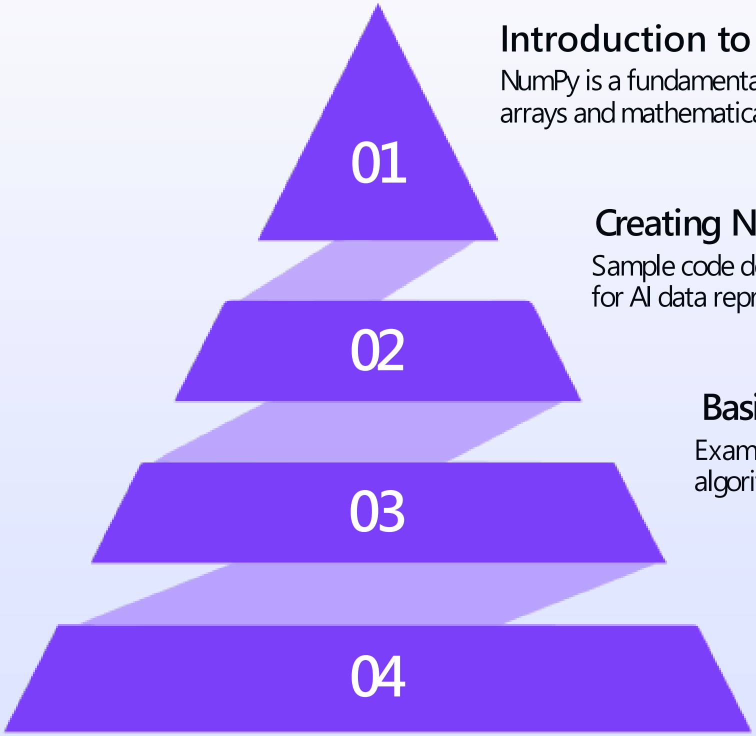
Break and Continue

Keywords that modify loop behavior, allowing early termination or skipping iterations when processing complex AI datasets.

02

Python Libraries for AI

NumPy and Mathematical Operations



Introduction to NumPy

NumPy is a fundamental Python library for numerical computing in AI, providing support for large multi-dimensional arrays and mathematical functions.

Creating NumPy Arrays

Sample code demonstrating array creation using np.array(), np.zeros(), np.ones(), and np.arange() functions for AI data representation.

Basic Array Operations

Examples of indexing, slicing, reshaping, and concatenating arrays to prepare data for AI algorithms and neural networks.

Mathematical Functions

Implementing mathematical operations like addition, subtraction, multiplication, and division on arrays with vectorized operations for efficiency.

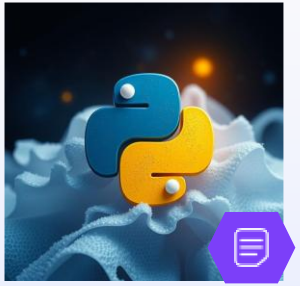
Pandas for Data Manipulation

Introduction to Pandas

Pandas is a Python library for data manipulation that provides data structures like DataFrame and Series for efficient data analysis in AI applications.

DataFrame Basics

~~~python



## Data Selection

```python

## Filtering Data

```python
~~~

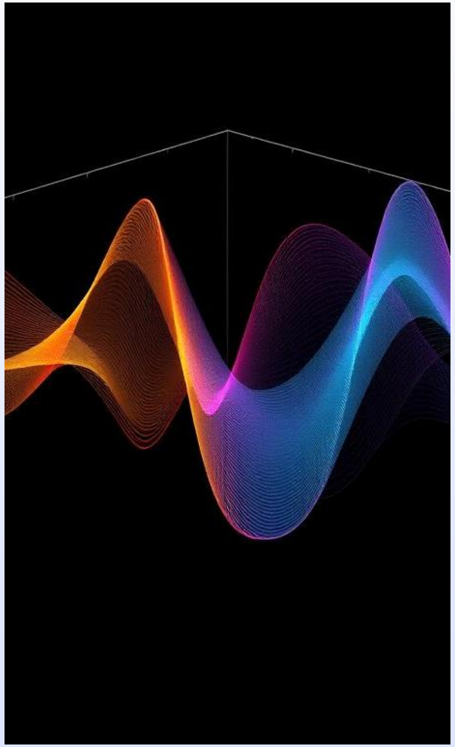
Matplotlib for Visualization

Introduction to Matplotlib

Matplotlib is a Python library for creating static, interactive, and animated visualizations, essential for data analysis in AI applications.

Basic Plotting Functions

Learn to create simple plots using plt.plot() for line graphs and plt.scatter() for scatter plots to visualize relationships in data.



Customizing Visualizations

Enhance plots with titles, labels, legends, and color schemes to improve readability and effectively communicate insights from AI models.

Multiple Plots

Create subplots using plt.subplots() to compare different datasets or model outputs side by side for comprehensive analysis.

03

Practical AI Programming Examples

Machine Learning Model Implementation

Machine Learning Libraries Overview

Introduction to essential Python libraries like scikit-learn, TensorFlow, and PyTorch that provide frameworks for implementing various machine learning models.

Data Preprocessing Techniques

Exploring Python code for data cleaning, normalization, and feature engineering using pandas and NumPy before model implementation.

Classification Models Implementation

Step-by-step implementation of classification algorithms including decision trees and support vector machines with sample code demonstrations.

Regression Analysis Code

Python implementation of linear and polynomial regression models with visualization using matplotlib and evaluation metrics calculation.

Neural Network Construction

Neural Network Basics

Neural networks are computational models inspired by the human brain, using interconnected nodes to process data and learn patterns.

Building with PyTorch

~~~python

### TensorFlow Implementation

```python

### Training Neural Networks

```python
~~~

Data Preprocessing Techniques

Data Cleaning

~~~python

### Feature Scaling

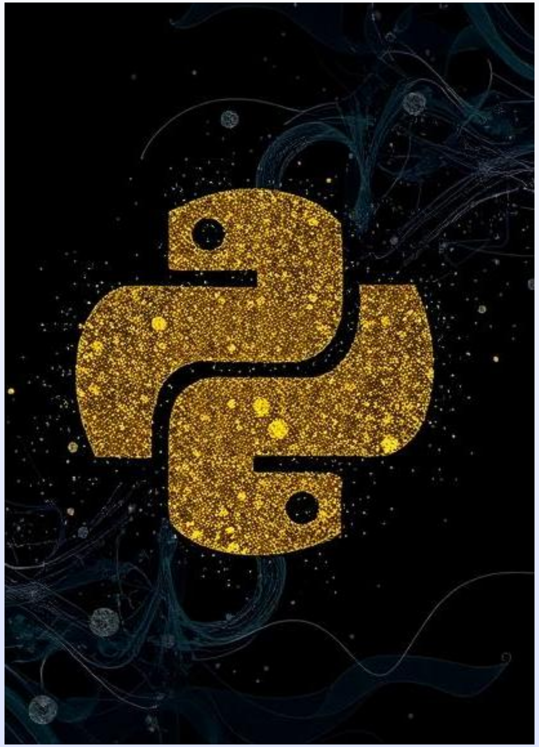
```python

### Encoding Categorical Data

```python

### Feature Selection

```python


~~~

Thanks ！