Unit I: Introduction to Data Science and Python

DSC 481 - Fundamentals of Data Science

Pokhara University

Faculty of Management Studies

3 Hours

Learning Objectives

- Explain what Data Science is and describe its applications
- Understand the Data Science workflow and process
- Set up Python environment for data science
- Use Jupyter Notebook and Google Colab
- Write basic Python programs with variables, data types, and operators
- Execute and save Python code in notebooks

What is Data Science?

- Interdisciplinary field
- Uses scientific methods, algorithms, systems, and statistics
- Extracts knowledge from structured and unstructured data
- Goal: actionable insights for decision-making

Data Science Applications

- **Healthcare:** Disease prediction, personalized medicine
- **Finance:** Fraud detection, credit risk assessment, market prediction
- **E-commerce:** Recommendations, customer segmentation, demand forecasting
- **Social Media:** Sentiment analysis, content recommendation

Data Science Workflow (CRISP-DM)

1. Business Understanding
2. Data Collection
3. Data Cleaning & Preparation
4. Exploratory Data Analysis
5. Modeling & Analysis
6. Evaluation & Validation
7. Deployment & Communication

Iterative process; may revisit stages

The Data Science Process

Stage	Example Activities	Example Tools
Collection	APIs, web scraping, databases	Python, Pandas
Cleaning	Missing values, outliers	Pandas, NumPy
Exploration	Patterns, visualization	Matplotlib
Modeling	Predictive models	Scikit-learn
Visualization	Charts, dashboards	Seaborn, Plotly
Deployment	Reporting, APIs	Dash, Streamlit

Roles in Data Science

- **Data Analyst:** analyzes data, reports, dashboards
- **Data Engineer:** builds pipelines, manages databases
- **Data Scientist:** develops ML models, advanced analytics, research

Why Python for Data Science?

- Easy to learn; readable syntax
- Rich ecosystem of libraries
- Large, active community
- Versatile; used in industry
- Open source and free

Key Python Data Science Libraries

- **NumPy:** numerical computing
- **Pandas:** data manipulation and cleaning
- **Matplotlib:** plotting and visualization
- **Seaborn:** statistical visualization
- **Scikit-learn:** machine learning algorithms
- **Plotly:** interactive dashboards & charts

Python Environment Setup

- **Anaconda Distribution** (recommended)
  - Python + 250+ data science packages
  - Conda package manager, Jupyter Notebook
- **Jupyter Notebook** (browser-based)
- **Google Colab** (cloud-based)

Anaconda Distribution

- Package management: Conda
- Environment management: isolated environments
- Pre-installed libraries: NumPy, Pandas, Matplotlib, etc.
- IDEs included: Jupyter Notebook, Spyder, VSCode

Install from anaconda.com

Jupyter Notebook Basics

- Interactive coding environment
- Combines code, text, visualizations
- Cells: run code and see output immediately
- Save notebooks as `.ipynb` files

Jupyter Notebook Interface

- **Menu bar:** File, Edit, View, Run
- **Toolbar:** quick actions
- **Code cells / Markdown cells**
- **Shortcuts:**
  - Shift+Enter : run cell
  - A/B : add cell above/below
  - DD : delete cell
  - M : markdown / Y : code

Google Colab Alternative

- No installation needed (browser only)
- Free GPU/TPU access, saves to Google Drive
- Easy sharing and collaboration

Go to colab.research.google.com

Python Basics: First Program

```
print("Hello, Data Science!")
```

Output: Hello, Data Science!

Check your Python version:

```
import sys
print("Python version:", sys.version)
```

Variables in Python

- Containers for data values

```
name = "Sita"
age = 21
gpa = 3.75
is_student = True
```

- Naming:
  - Start with a letter or underscore
  - Case-sensitive
  - Cannot use Python keywords
  - Use descriptive names

Python Data Types (Primitives)

Type	Example
int	`42`
float	`3.14`
str	`"Hello"`
bool	`True`, `False`
NoneType	`None`

Check type:

```
age = 25
print(type(age))
```

Python Arithmetic Operators

- `+` (Addition)
- `-` (Subtraction)
- `*` (Multiplication)
- `/` (Division)
- `//` (Floor division)
- `%` (Modulus, remainder)
- `**` (Exponentiation)

Arithmetic Operator Examples

```
a = 10
b = 3
print(a + b)      # 13
print(a / b)      # 3.333...
print(a // b)     # 3
print(a % b)      # 1
print(a ** b)     # 1000
```

Comparison & Logical Operators

- Comparison:
  - `==`, `!=`, `<`, `>`, `<=`, `>=`
- Logical:
  - `and`, `or`, `not`

Comparison Example

```
x = 5
y = 10
print(x == y)        # False
print(x < y)         # True
print((x != y) and (x < y)) # True
```

Type Conversion

- Convert types:
  - `int()`, `float()`, `str()`

```
age = int("25")
price = float("19.99")
score = str(95)
```

String Formatting & Operations

- Concatenate: `"Ram" + " Sharma"`
- f-strings: `f"{name} is {age}"`
- Indexing: `"Python"[0]`

String Example

```
first_name = "Ram"
last_name = "Sharma"
full_name = first_name + " " + last_name
print(full_name)

message = f"{first_name} is a student"
print(message)

word = "Python"
print(word[0])      # P
print(word[-1])     # n
```

Built-in Functions

- `print()`, `type()`, `len()`, `abs()`, `round()`, `max()`, `min()`

```
print(type(42))
print(len("Python"))
print(abs(-10))
print(round(3.7))
```

Live Demo

- Open Jupyter/Colab
- Create notebook ("Unit_I_Practice")
- Run code cells and markdown documentation
- Create variables, operations, save & export

Hands-On Exercise #1

- Print welcome message
- Create and print variables
- Arithmetic operations
- Type conversion

Hands-On Exercise #2

- Boolean expressions
- String operations
- Use built-in functions

Common Beginner Mistakes

- Indentation errors
- Mixing types (e.g., `"Age: " + 21` fails)
- Using keywords as variables (`class = "Data Science"`)
- Case sensitivity (`Name` vs `name`)

Knowledge Check

- List three stages of DS workflow
- Name three Python libraries for DS
- Data type of `3.14`?
- What does `//` do?
- Convert `"100"` to integer?

Unit Summary & Takeaways

- DS concepts and workflow
- Python ecosystem and setup
- Basic Python: variables, data types, operators
- Jupyter/Colab coding: practical skills

Homework & Next Steps

- Finish exercises
- BMI calculator & "About Me" program
- Read "Python Crash Course" Chapters 1-2
- Explore docs and tutorials
- Next: Python Programming Basics & Operators

Questions & Discussion

- Any questions?
- Office hours: [Your availability]
- Contact: [Your email]
- Practice daily—code every day

Thank You!

> "The only way to learn a new programming language is by writing programs in it."
> — Dennis Ritchie

See you next class!

Next topic: Python Programming Basics & Operators

Don't forget: Submit homework by [deadline]