ScHARe

What are Think-a-Thons?

BE A PART OF THE FUTURE
OF KNOWLEDGE GENERATION

Think-a-Thons (TaT)

- Monthly sessions (2 1/2 hours)
- Instructional/interactive
- Designed for new and experienced users
- Research & analytic teams to:
  - Conduct health disparities, health outcomes, bias mitigation research
  - Analyze/create tools for bias mitigation
- Publications from research team collaboration
- Networking
- Mentoring and coaching
- Focus:

Types:

- ✓ **Instructional / Tutorial**
- ✓ **Collaborative Research Teams**
- ✓ **Bias mitigation**

ScHARe

Think-a-Thon

Artificial Intelligence and Cloud Computing Basics

Terra: Datasets and Analytics

Register:

bit.ly/think-a-thons

Think-a-Thon Instructional Tutorials

Web: bit.ly/think-a-thons

February	**Artificial Intelligence and Cloud Computing 101**
March	**ScHARe 1 – Accounts and Workspaces**
April	**ScHARe 2 – Terra Datasets**
May	**ScHARe 3 – Terra Google-hosted Datasets**
June	**ScHARe 4 – Terra ScHARe-hosted Datasets**
July	**An Introduction to Python for Data Science – Part 1**
August	**An Introduction to Python for Data Science – Part 2**
September	**ScHARe 5: A Review of the ScHARe Platform and Data Ecosystem**
October	**Preparing for AI 1: Common Data Elements and Data Aggregation**
November	**Preparing for AI 2: An Introduction to FAIR Data and AI-ready Datasets**
January	**Preparing for AI 3: Computational Data Science Strategies 101**
February	**Preparing for AI 4: Overview Prep for AI Summary with Transparency, Privacy, Ethics**

ScHARe for Educators (Community Colleges & Low Resource MSIs)
ScHARe for American Indian / Alaska Native Researchers
ScHARe for Non-Biomedical Researcher Coders and Programmers to conduct Research

The monthly **ScHARe Think-a-Thons** scheduled or archived below are designed so participants reach one of these goals (as noted with each session):

- **Goal 1:** Achieve a **better understanding of both the fields and the terminology** used to describe the AI/cloud computing infrastructure, components and processes.
- **Goal 2: Develop research questions and projects relevant to AI and cloud computing** that leverage the cutting-edge technology and data/computing resources now available to health disparities researchers (including the ones at their disposal on the ScHARe platform).

Upcoming Think-a-Thons | Past Think-a-Thons | See FAQs

Think-a-Thon Schedule

Think-a-Thons are held on the third Wednesday of each month. Accommodations information | Think-a-Thon recordings

Date	Time	Topic	Register
March 20, 2024	2:00 – 4:30 p.m. ET	**Preparing for AI-driven Research on ScHARe: A Comprehensive Review and Brainstorming Session – Part 2** Toward Goal 2: Prepares participants for ScHARe research collaborations by covering: • Choosing computational strategies (AI, ML, statistics) • An overview of Python data science libraries • The significance of testing and monitoring in algorithm development • The role of open science in ensuring reproducible and transparent AI-based research For researchers and students at all levels who want to collaborate on ScHARe to develop innovative and publishable research projects	Register *Registration closes at 12:00 p.m. ET on the day of the event.*

ScHARe Think - a - Thons

PAST Think-a-Thons Posted

November 15, 2023	2.5 hours	View video: **Preparing for AI 2: An Introduction to FAIR Data and AI-ready Datasets** Toward Goal 1: How to prepare an AI-ready dataset using gold standard data management principles, including: • Making datasets findable, accessible, interoperable, and reusable (FAIR) • Using transparent data documentation to foster data re-use • Ensuring that selected data addresses expected outcomes and drives meaningful AI insights • Handling missing data through strategies, proxies, and synthetic data	View slides (PDF, 4 MB)

https://www.nimhd.nih.gov/resources/schare/think-a-thons.html

Upcoming

ScHARe

Think-a-Thons (TaT)

Research Teams

Title: Data Science Projects 1 – Health Disparities and Individual SDoH

Description: Exploring the impact of individual Social Determinants of Health on health outcomes: a hands-on session for researchers and students at all levels interested in collaborating on ScHARe to develop innovative research questions and projects leading to publications.

Title: Data Science Projects 2 - Health Disparities and Structural SDoH

Description: Assessing the impact of structural Social Determinants of Health on health outcomes: a hands-on session for researchers and students at all levels interested in collaborating on ScHARe to develop innovative research questions and projects leading to publications.

Title: Data Science Projects 3 – Health Outcomes

Description: Investigating the influence of non-clinical factors on disparities in health care delivery: a hands-on session for researchers and students at all levels interested in collaborating on ScHARe to develop innovative research questions and projects leading to publications.

- Foster a research paradigm shift to use Big Data
- Promote use of Dark Data
- Generational Career & Discipline Exchange

- Multi-career (students to sr. investigators)
- Multi-discipline (data scientist & researchers)
- Feature Datasets with Guest Expert Leads
- Secure experts in topic area, analytics, data sources etc. to provide guidance
- Generate research idea - decide potential design, datasets & analytics
- Select co-leads to coordinate completion outside of TaT
- Publications

Register:

bit.ly/think-a-thons

Generational Career & Discipline Exchange

Research Think - a - Thons

Expectation of the Research Project.

- The launch of the project will occur during the Think-a-Thon.
  - Pre-Assigned Co-Leads: Data Science Expert and a Health Disparity/Health Care Delivery Expert
  - There will be 4 sessions: 2 python, 1 R and 1 Statistic defined research collaborative
  - Volunteers who want to participate in health disparity/health care delivery research will select one of the 4 sessions based upon the analytics expected to be used
  - In the breakouts, the group will decide the research topic and which data sets will be used.

- The co-leads will assign tasks to the participants for the next **three months** to complete the project in preparation for publication. There will be meetings other than Think-a-Thons to:
    - review progress of tasks
    - help/teach others what each participant is contributing
    - assessing what else needs to be completed

https://www.nimhd.nih.gov/resources/schare/think-a-thons.html

Research Think - a - Thons

During Think-a-Thon

- ScHARe Terra Workspace (Data Co-lead is primary to create and to monitor workspace collaborators
- Research Topic (Science co-lead will guide the discussion to the consensus topic)
- Likely data sets to be used for topic

Project Expectations:

- Literature review
- Data set assessment for AI readiness (i.e. complete variables needed for project, fair representation of populations, missing variables, incompleteness of variables, data gaps, etc)
- Data Dictionary
- Data Sheet and Data Set Facts
- Design description to ensure that the outcome expected is probable.
- Decision on analytics and training to be used (complete a methodology description, including a model card)
- Test results for biases (document the types of biases encountered and how each addressed)
- Draft Publication

https://www.nimhd.nih.gov/resources/schare/think-a-thons.html

ScHARe

Research Think - a - Thons

Experience Conducting Ethical AI

TRANSPARENCY:

- Def:
  - Public Perception & understanding of how AI works
  - Comprehend the algorithmic views and decisions taken based on them
- Technical Documentation for duplication / re-use
- Tools:
  - Data Dictionary
  - Health Sheet (Data Sheet)
- Model Cards (capabilities & purpose of algorithms are openly and clearly communicated to relevant stakeholders)
- Documentation of methodologies
- Doesn't disclose intellectual property

FAIRNESS:

- **Findable**: providing metadata, documentation, and clear identifiers
- **Accessible**: wide audience
- **Interoperable**: standardized formats and APIs enable seamless integration.
- **Reusable:** clear documentation, licensing, reduce redundancy

Metadata and data should be easy to find for both humans and computers

Ensure that data represents relevant populations

https://www.nimhd.nih.gov/resources/schare/think-a-thons.html

Think-a-Thons Training/Mentoring Pipeline

NLM
OIC Experts
Fellows

Think-a-Thons

✓ **Instructional**
✓ **Research**

N3C
Aim AHEAD
BioData Catalyst
All of US
AnVil
HEAL

Using experts + *To train and mentor novice users* + *To increase diverse perspectives in biomedical research*

Goal: "Upskilling"

- ✓ Data science specialist into health disparities and health outcomes research
- ✓ Health Disparity/Outcomes researchers into using big data and cloud computing

Target Audience:

- ✓ Underrepresented populations (women, race/ethnic) users not trained in data science
- ✓ Data scientist with no or little research experience.
- ✓ Resource & Tool for Community Colleges and Low Resource MSIs and Organizations

Interest poll

I am interested in (check all that apply):

- ☐ Learning about Health Disparities and Health Outcomes research to apply my data science skills
- ☐ Conducting my own research using AI/cloud computing and publishing papers
- ☐ Connecting with new collaborators to conduct research using AI/cloud computing and publish papers
- ☐ Learning to use AI tools and cloud computing to gain new skills for research using Big Data
- ☐ Learning cloud computing resources to implement my own cloud
- ☐ Developing bias mitigation and ethical AI strategies
- ☐ Other

ScHARe

Computational Strategies

Data Science Computational Strategies

Choosing Between Traditional Statistics and AI/ML

- Welcome to the **second part** of our workshop on conducting research projects

- Today's **overview** will cover selecting computational strategies in data science. We'll explore the decision-making process involved in choosing between traditional statistics and Artificial Intelligence (AI)/Machine Learning (ML)

- We will help you understand the **fundamental differences** between these approaches and their respective advantages and disadvantages, and point you to helpful **Python libraries** for each strategy

Decision-Making Process

Choosing the **right approach to analyzing data** is critical for achieving research objectives effectively

The decision-making process in selecting computational strategies involves several **key steps**:

1. We must clearly **define the research problem** or question we aim to address
2. Next, we must consider the **nature of the data we have**, our **research goals**, and the **resources available** to us

Based on these factors, we then choose the most appropriate computational strategy

Traditional Statistics vs. AI and ML

Traditional Statistics and modern computational techniques such as Artificial Intelligence (AI) and Machine Learning (ML) offer **distinct approaches to data analysis:**

- **Traditional Statistics** focuses on **hypothesis testing, inference, and the application of parametric and non-parametric methods.** It emphasizes **interpretability, reliance on assumptions, and limitations in handling complex datasets**

- In contrast, **AI and ML** prioritize **pattern recognition, prediction, and the development of predictive models.** They emphasize **scalability, complexity management, and the ability to process large volumes of data efficiently**. However, AI and ML models often **trade interpretability for increased predictive power**, leading to challenges in understanding their decision-making processes

Analyses Enabled by AI and Big Data

- Artificial Intelligence (AI) and Big Data enable a **wide range of advanced analyses that go beyond traditional statistical methods**, including:
  - **predictive analytics**
  - **natural language processing**
  - **image recognition**
  - **anomaly detection** (identifying unusual patterns or data points that deviate significantly from the expected norm)

- AI and Big Data empower researchers to extract **valuable insights from vast and complex datasets**, leading to more accurate predictions, enhanced decision-making and problem-solving capabilities

- Examples of AI and Big Data analyses include predictive modeling for **disease outbreak prediction** and **sentiment analysis** of social media data for public health monitoring

Conclusion and Recommendations

- It's important to consider the **nature of the research problem**, the **characteristics of the data,** and the **desired outcomes** when choosing a computational strategy

- **Recommendations:**
  1. Assess the **research objectives and data characteristics** before selecting a strategy
  2. Leverage the **strengths of each approach** to maximize the insights gained from data analysis
  3. Stay informed about **emerging technologies and methodologies** in data science to adapt to evolving research needs

- By making **informed decisions** about computational strategies, researchers can enhance the quality and impact of their research outcomes

ScHARe

Traditional Statistics

Overview

- **Strengths:** robust, interpretable, well-established methodologies
- **Weaknesses:** limited predictive power, assumption-dependent, often focused on hypothesis testing
- **Data types & use cases:** numerical data, identifying trends, correlations, causal relationships

- **Popular Python libraries:** NumPy, SciPy, Pandas

1. Descriptive Statistics

- **Strategy:** Summarizing and describing key features of healthcare data, such as mean, median, standard deviation, and percentiles

- **Applications:** Understanding the central tendency and variability in healthcare variables

- **Python Libraries:** NumPy, pandas

2. Inferential Statistics

- **Strategy:** Making predictions or inferences about a population based on a sample from that population

- **Applications:** Drawing conclusions about healthcare disparities from a subset of relevant data

- **Python Libraries:** SciPy, statsmodels

3. Hypothesis Testing

- **Strategy:** Evaluating statistical significance to determine whether observed differences are likely to be real or due to chance

- **Applications:** Testing hypotheses about healthcare interventions or disparities

- **Python Libraries:** SciPy, statsmodels

4. Analysis of Variance (ANOVA)

- **Strategy:** Assessing the statistical significance of differences among group means in healthcare data

- **Applications:** Comparing means across multiple categories to identify significant differences

- **Python Libraries:** SciPy, statsmodels

5. Chi-Square Test

- **Strategy:** Assessing the association between categorical variables in healthcare datasets

- **Applications:** Examining relationships between demographic factors and health outcomes

- **Python Libraries:** SciPy, pandas

6. Regression Analysis

- **Strategy:** Modeling the relationship between dependent and independent variables in healthcare data

- **Applications:** Predicting health outcomes based on various factors, identifying disparities

- **Python Libraries:** Statsmodels, scikit-learn

7. Survival Analysis

- **Strategy:** Analyzing time-to-event data, such as the time until a patient experiences a particular health event

- **Applications:** Studying disparities in disease progression or survival rates

- **Python Libraries:** Lifelines, statsmodels

8. Correlation Analysis

- **Strategy:** Examining the strength and direction of relationships between two continuous variables in healthcare datasets

- **Applications:** Assessing associations between risk factors and health outcomes

- **Python Libraries:** NumPy, pandas

9. Logistic Regression:

- **Strategy:** Modeling the probability of a binary outcome in healthcare data
- **Applications:** Analyzing factors influencing the likelihood of specific health events
- **Python Libraries:** Statsmodels, scikit-learn

10. Bayesian Statistics

- **Strategy:** Updating beliefs about parameters based on new evidence in a probabilistic framework

- **Applications:** Incorporating prior knowledge into healthcare disparities research

- **Python Libraries:** PyMC3, Stan

11. Time Series Analysis

- **Strategy:** Analyzing temporal patterns and trends in healthcare data
- **Applications:** Studying disparities over time in health outcomes or interventions
- **Python Libraries:** Statsmodels, Pandas

ScHARe

Artificial Intelligence and Machine Learning

Main AI Computational Strategies

- Artificial Intelligence (AI) encompasses various computational strategies aimed at **mimicking human intelligence**

- These strategies are implemented using **different algorithms and techniques**

- **Python,** being a versatile language, offers numerous libraries for implementing AI strategies effectively

Machine Learning (ML)

- Machine learning involves the development of algorithms that **enable computers to learn from and make predictions or decisions based on data.** ML allows computers to improve at a specific task without explicit programming, by learning from data

- Examples:
  - Linear Regression
  - Decision Trees
  - Random Forest

- Commonly Used Python Libraries:
  - scikit-learn
  - TensorFlow
  - Keras
  - PyTorch

Deep Learning

- Deep learning is a subset of machine learning that **is inspired by the structure and function of the brain.** It uses **artificial neural networks** comprising multiple layers to **learn complex patterns from data**

- Examples:
  - Convolutional Neural Networks (CNNs) for image recognition
  - Recurrent Neural Networks (RNNs) for sequence data
  - Generative Adversarial Networks (GANs) for generating synthetic data

- Commonly Used Python Libraries:
  - TensorFlow
  - Keras
  - PyTorch

Natural Language Processing (NLP)

- NLP involves the interaction between computers and humans using natural language. It focuses on giving **computers the ability to understand and manipulate human language**

- Examples:
  - Sentiment Analysis
  - Named Entity Recognition (NER) (it categorizes specific elements within text)
  - Machine Translation

- Commonly Used Python Libraries:
  - NLTK (Natural Language Toolkit)
  - spaCy
  - Transformers

Reinforcement Learning

- Reinforcement learning focuses on training agents to make sequential decisions by interacting with an environment

- Examples:
  - Game playing (e.g., AlphaGo)
  - Robotics control
  - Recommendation systems

- Commonly Used Python Libraries:
  - OpenAI Gym
  - TensorFlow Agents
  - Stable Baselines

Evolutionary Algorithms

- Evolutionary algorithms are inspired by biological evolution and involve **optimization techniques** based on natural selection and genetic variation. Specifically, they **mimic natural selection** to solve problems by iteratively **refining populations of candidate solutions**

- Examples:
  - Genetic Algorithms
  - Genetic Programming
  - Evolutionary Strategies

- Commonly Used Python Libraries:
  - DEAP (Distributed Evolutionary Algorithms in Python)
  - PyGMO (Python Parallel Global Multiobjective Optimizer)

Quiz 1

Scenario: You are a public health researcher investigating the factors contributing to higher rates of heart disease among a specific minority population in your community. You have a dataset containing information about thousands of individuals, including demographics, socioeconomic factors, health history, and access to healthcare.

Question: Which approach would be most suitable for analyzing this data to understand the disparities in heart disease rates?

a) **Traditional Statistics:** Calculate average income levels and compare them to heart disease prevalence across different zip codes within the community.

b) **Machine Learning:** Develop a machine learning model to predict the likelihood of developing heart disease for individuals based on their data.

c) **Both Traditional Statistics and Machine Learning:** Use traditional statistics to explore initial relationships and then build a machine learning model to identify complex patterns contributing to the disparities.

Quiz 1

Answer: (C) Both Traditional Statistics and Machine Learning

Explanation:

- Traditional statistics can reveal basic trends, like correlations between income and heart disease prevalence across zip codes. This can provide initial clues about potential disparities.
- Machine learning can be powerful in health disparities research. It can analyze complex interactions between various factors (e.g., income, access to healthcare, environmental factors) and their combined influence on heart disease risk within the specific population.
- By combining traditional statistics for initial exploration with machine learning for in-depth analysis, you gain a comprehensive understanding of the factors contributing to the observed health disparities.

ScHARe

Python Libraries

What is Python?

Python is a **computer programming language** used in data science to:

- manipulate and analyze data
- create data visualizations
- build machine learning algorithms

Imagine you want to tell your computer what to do, by giving it clear, easy-to-understand commands. That's what Python is like!

- **Easy to learn:** Python uses words and phrases that are close to everyday English, making it a good choice for beginners
- **Versatile:** You can use Python for many things
- **Free and open-source:** Anyone can use and improve Python for free: there's a large and helpful community to answer your questions
- **Popular:** there are lots of online resources to help you learn

Why Python?

According to SlashData:

- there are 8.2 million Python users
- **69%** of machine learning developers and data scientists **use Python (vs. 24% using R)**

Source
stackify.com/learn-python-tutorials/

How to learn Python

How long does it take to learn Python?

It can take **2 to 5 months**, but you can write your first short program in **minutes**

Can you learn Python with no experience?

Python is the **perfect** programming language **for people without any coding experience**, as it has a simple syntax and is very accessible to beginners

Links to additional **free learning resources** will be provided

Introduction to Python Data Science Libraries

- Python offers a **rich ecosystem of libraries** for data science tasks
- In this section, we'll introduce some of the most commonly used Python libraries in data science
- **Each library serves specific functions** in the data science workflow

What is a Python library?

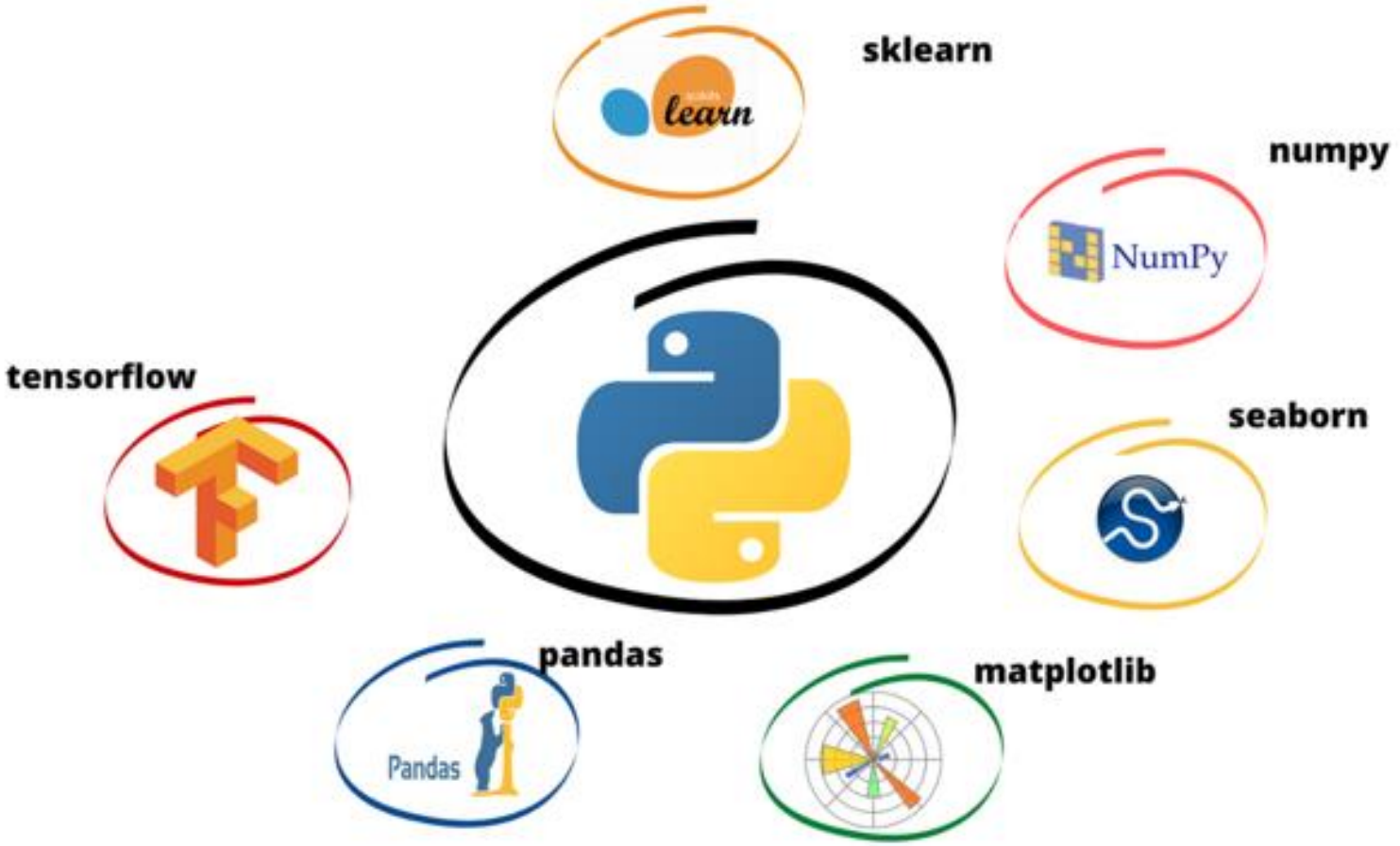
It's like a **collection of tools or functions** that someone else has **already built and packaged up** for you to use in your own programs

When you're writing a Python program and you need to do something specific, like create visualizations, you can often find a library that **already has the tools you need for that job**

You just need to **"import" the library** into your program, and you can start using its tools right away

Overview of Python Data Science Libraries

Python data science libraries are essential for **data manipulation, analysis, and visualization** tasks.



NumPy: The Foundation for Numerical Computing

Overview

A fundamental package for scientific computing, providing support for large, multi-dimensional arrays (ordered collections of items) and matrices

Characteristics

- Provides efficient **multidimensional arrays** for data storage and manipulation
- Enables **mathematical operations** on large datasets
- Lays the **groundwork for data analysis** with other libraries

Example application

Calculating statistical measures such as mean, median, and standard deviation of health indicators (e.g., life expectancy) across various demographic groups

SciPy: Extending Computing Capabilities

Overview

An open-source library that builds on NumPy and provides additional functionality for mathematical and scientific computing

Characteristics

- Offers **advanced algorithms** for scientific computing beyond NumPy
- Includes **tools for optimization**, integration, and signal processing
- **Complements NumPy** for diverse scientific computing tasks

Example application

Conducting hypothesis testing to evaluate the effectiveness of interventions aimed at reducing health disparities, such as comparing pre- and post-intervention health indicators

Pandas: Wrangling Data Like a Pro

Overview

A powerful library for data manipulation and analysis, offering data structures and functions for manipulating structured data

Characteristics

- Offers powerful data structures like **DataFrames** for handling tabular data
- Enables **data cleaning, manipulation, and exploration** with ease
- **Integrates** seamlessly with other data science libraries

Example application

Exploring correlations between socio-economic factors (e.g., income, education level) and health outcomes (e.g., mortality rates)

Matplotlib: Visualizing Insights

Overview

A comprehensive library for creating static, animated, and interactive visualizations in Python, offering a wide range of plotting functions

Characteristics

- Creates a wide variety of static, animated, and interactive **visualizations**
- Enables **customization** for clear and compelling data storytelling
- Integrates with other libraries for comprehensive **data exploration**

Example application

Creating visualizations such as bar charts or pie charts to illustrate disparities in healthcare access among different ethnic or socio-economic groups

Seaborn: Building on Matplotlib for Stats

Overview

A statistical data visualization library based on Matplotlib, providing a high-level interface for creating informative and attractive visualizations

Characteristics

- Offers a high-level interface built upon Matplotlib for statistical graphics
- Creates aesthetically pleasing and informative visualizations
- Ideal for exploring relationships and distributions within your data

Example application

Creating box plots or violin plots to compare distributions of health indicators (e.g., blood pressure levels) among different population segments

Scikit-learn: Machine Learning Made Accessible

Overview

A machine learning library that offers simple and efficient tools for data mining and data analysis, including classification, regression, clustering, and dimensionality reduction

Characteristics

- Provides a comprehensive library for various **machine learning** algorithms
- Enables tasks like **classification, regression, and clustering**
- Facilitates **model building, evaluation, and deployment**

Example application

Implementing machine learning algorithms to classify patients into different risk categories based on socio-economic factors and predict healthcare outcomes (e.g., hospital readmissions)

Statsmodels: Diving Deeper into Statistical Analysis

statsmodels

Overview

A library for estimating statistical models and conducting statistical tests, providing a wide range of statistical techniques

Characteristics

- Provides a collection of tools for **statistical modeling and econometrics**
- Enables robust **hypothesis testing**, estimation, and model selection
- Ideal for **in-depth statistical analysis** of health disparities data

Example application

Exploring correlations between socio-economic factors (e.g., income, education level) and health outcomes (e.g., mortality rates)

TensorFlow: Building Powerful Deep Learning Models

Overview

An open-source machine learning framework developed by Google, widely used for building and training deep learning models

Characteristics

- Open-source framework for numerical computation **and large-scale machine learning**
- Particularly adept at **deep learning**, a powerful subset of machine learning
- Enables building and training **complex models** for tasks like natural language processing

Example application

Training convolutional neural networks (CNNs) to analyze medical images (e.g., X-rays, MRIs) and detect signs of disease or abnormalities associated with health disparities

PyTorch: A Powerful Deep Learning Framework

PyTorch

Overview

Provides support for distributed training across multiple GPUs and devices, enabling researchers to train large-scale machine learning models efficiently

Characteristics

- Well-suited for **rapid prototyping and experimentation**
- **User-friendly** approach that lowers the barrier to entry for deep learning
- PyTorch models can be efficiently deployed in production environments

Example application

Adapting pre-trained language models for healthcare-specific NLP tasks, such as extracting information about social determinants of health from unstructured text data

Quiz 2

Which Python library is commonly used for data manipulation and analysis, offering data structures and functions for working with structured data?

a) NumPy

b) Pandas

c) SciPy

d) Statsmodels

Quiz 3

Which Python library is known for its statistical data visualization capabilities and is based on Matplotlib?

a) NumPy

b) Pandas

c) Seaborn

d) TensorFlow

Example Application with Code (NumPy)

python Copy code

```python
import numpy as np

# Sample health outcome data for different demographic groups
health_outcomes = np.array([
    [120, 80, 100],  # Group 1
    [90, 110, 95],   # Group 2
    [100, 95, 105]   # Group 3
])

# Calculate mean, median, and standard deviation
mean_outcomes = np.mean(health_outcomes, axis=1)
median_outcomes = np.median(health_outcomes, axis=1)
std_outcomes = np.std(health_outcomes, axis=1)

print("Mean outcomes:", mean_outcomes)
print("Median outcomes:", median_outcomes)
print("Standard deviation of outcomes:", std_outcomes)
```

NumPy simplifies numerical computations and array operations in Python

Example: Calculating **summary statistics** for health outcome data and detecting variations across demographic groups

Libraries in notebooks

A **Jupyter Notebook** is an interactive analysis tool that includes:

- **code cells** for manipulating and visualizing data in real time (Terra notebooks support **Python or R**)
- **documentation** to make it easier to share and reproduce your analysis

In past Think-a-Thons, we:

- covered the basics of **creating your first notebook**
- **explored the instructional notebooks** available in the ScHARe workspace

If you are not familiar with **programming**, the code in our notebooks is very easy to understand and reuse, and our tutorials will help you understand how notebooks work.

Why use notebooks?

A notebook integrates code and its output into a single document where you can run code, display the output, and also add explanations, formulas, and charts

Using notebooks:

- **is now a major part of the data science workflow** at research institutions across the globe
- can make your work **more transparent, understandable, repeatable, and shareable**
- will **speed up your workflow** and make it easier to communicate and share your results

ScHARe notebooks

Take a look at what a notebook can do by checking out the instructional notebooks that **ScHARe offers to help novice users** learn how to use the workspace and its resources

A list of the available notebooks is provided on the right.

List of ScHARe instructional notebooks

- **00_List of Datasets Available on ScHARe**: a list of the datasets available in the ScHARe Datasets collection.
- **01_Introduction to Terra Cloud Environment**: an introduction to the Terra platform and cloud environment.
- **02_Introduction to Terra Jupyter Notebooks**: an introduction to Jupyter Notebooks on the Terra platform.
- **03_R Environment setup**: instructions on how to setup your cloud environment for R-based notebooks.
- **04_Python 3 Environment setup**: instructions on how to setup your cloud environment for Python 3-based notebooks.
- **05_How to access plot and save data from public BigQuery datasets using R**: instructions on how to access, plot, and save data from datasets available on the cloud through the Google Cloud Public Datasets Program, using R.
- **06_How to access plot and save data from public BigQuery datasets using Python 3**: instructions on how to access, plot, and save data from datasets available on the cloud through the Google Cloud Public Datasets Program, using Python 3.
- **07_How to access plot and save data from ScHARe hosted datasets using Python 3:** instructions on how to access, plot, and save data from datasets hosted by ScHARe in this workspace.
- **08_How to upload access plot and save data stored locally using Python 3:** instructions on how to import to Terra, access, plot, and save data from datasets stored locally on your computer.

ScHARe

Python External Resources

Python resources

You can take advantage of the dozens of “**Python for data science” online tutorials** for beginners and advanced programmers listed here:

- Stackify - 30+ Tutorials to Learn Python
- FreeCodeCamp - Code Class for Beginners
- Harvard – Free Python Course
- Coursera – Free and Paid Python Courses
- LearnPython – Free Interactive Python Tutorials
- BestColleges – 10 Places to Learn Python for Free

Python resources

Stackify

30+ tutorials to learn Python

Top 30 Python Tutorials

In this article, we will introduce you to some of the best **Python tutorials.** These tutorials are suited for both beginners and advanced programmers. With the help of these tutorials, you can learn and polish your coding skills in Python.

1. Udemy
2. Learn Python the Hard Way
3. Codecademy
4. Python.org
5. Invent with Python
6. Pythonspot
7. AfterHoursProgramming.com
8. Coursera
9. Tutorials Point
10. Codementor
11. Google's Python Class eBook
12. Dive Into Python 3
13. NewCircle Python Fundamentals Training
14. Studytonight
15. Python Tutor
16. Crash into Python
17. Real Python
18. Full Stack Python
19. Python for Beginners
20. Python Course
21. The Hitchhiker's Guide to Python!
22. Python Guru
23. Python for You and Me
24. PythonLearn
25. Learning to Python
26. Interactive Python
27. PythonChallenge.com
28. IntelliPaat
29. Sololearn
30. W3Schools

Python resources

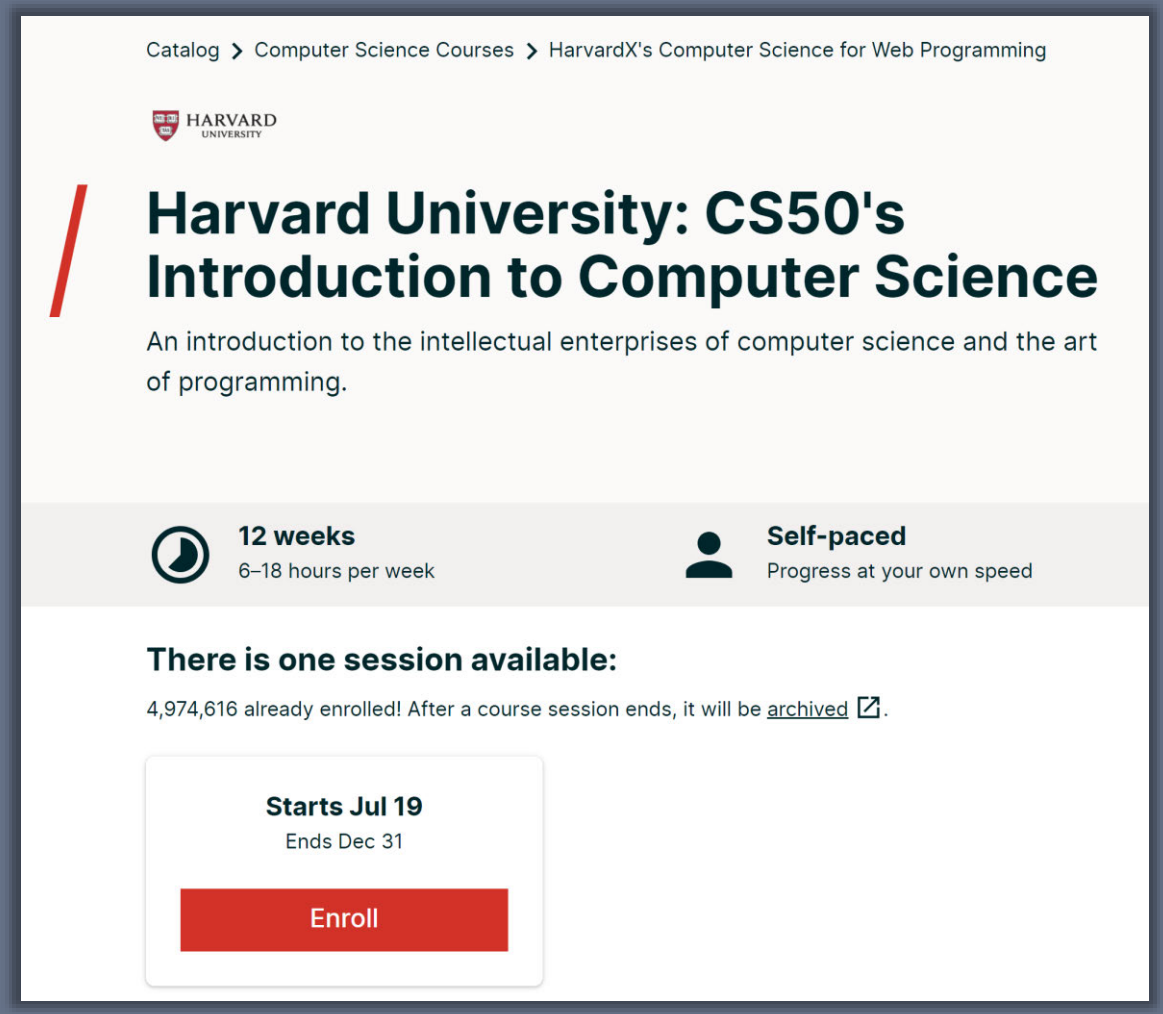
FreeCodeCamp

Code class for beginners

Python resources

Harvard

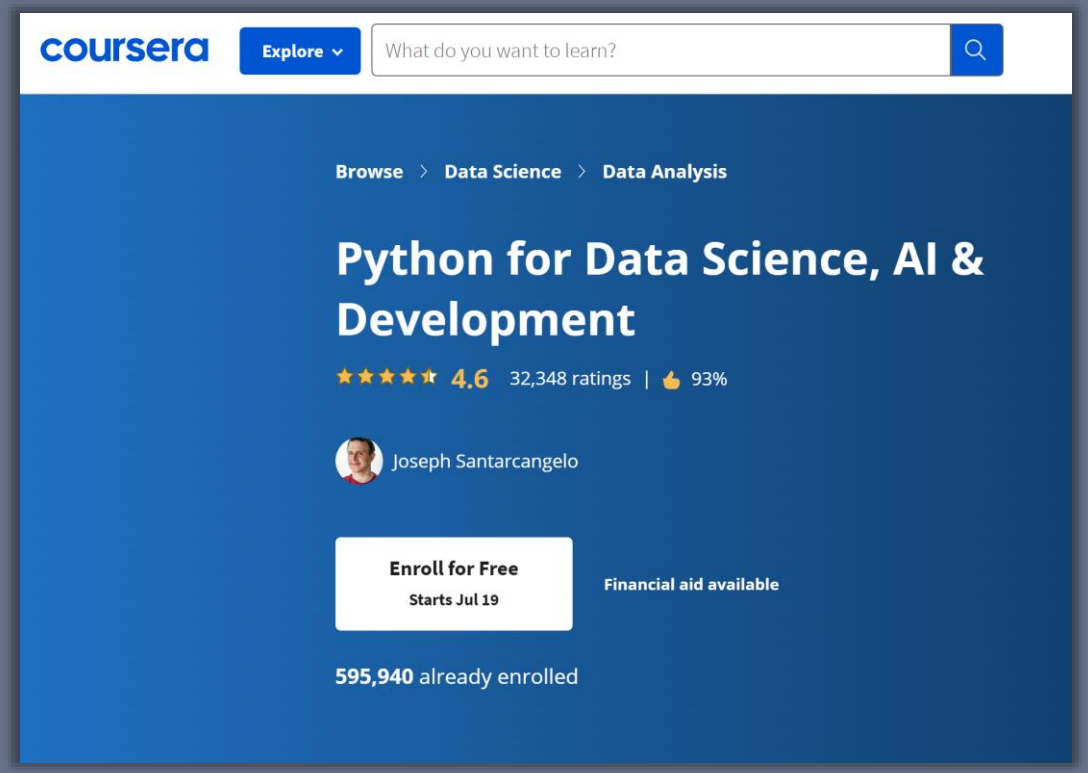
Free Python course



Python resources

Coursera

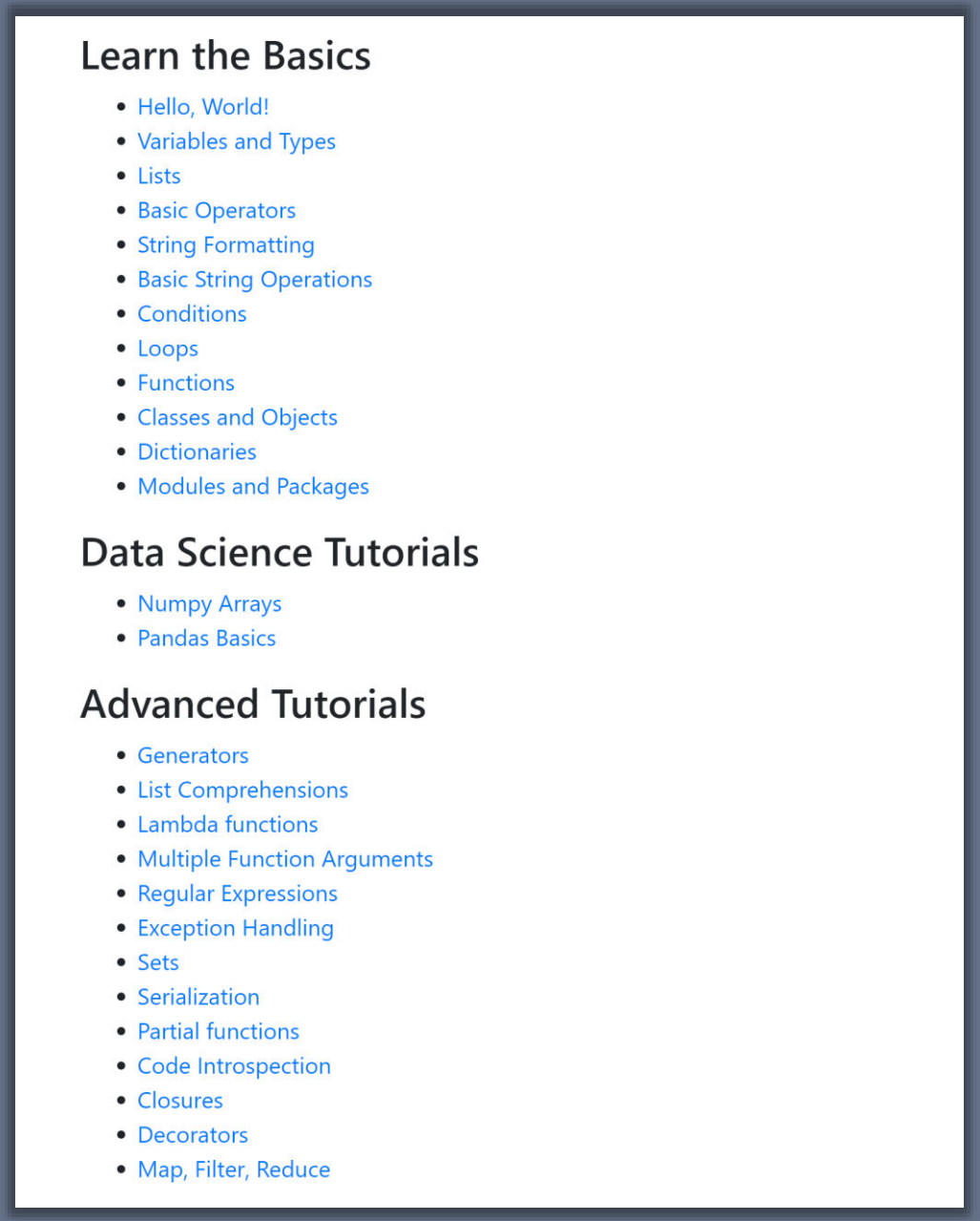
Free and paid Python courses



Python resources

LearnPython

Free interactive Python tutorials



Learn the Basics

- Hello, World!
- Variables and Types
- Lists
- Basic Operators
- String Formatting
- Basic String Operations
- Conditions
- Loops
- Functions
- Classes and Objects
- Dictionaries
- Modules and Packages

Data Science Tutorials

- Numpy Arrays
- Pandas Basics

Advanced Tutorials

- Generators
- List Comprehensions
- Lambda functions
- Multiple Function Arguments
- Regular Expressions
- Exception Handling
- Sets
- Serialization
- Partial functions
- Code Introspection
- Closures
- Decorators
- Map, Filter, Reduce

Python resources

BestColleges

10 places to learn Python for free