A Training on Python for Data Analysis

From Fundamentals to Machine Learning

Nahiyan Bin Noor, MS

July 22, 2025

Data Analyst - Intermediate
Institute for Digital Health & Innovation
University of Arkansas for Medical Sciences

Outline

Introduction to Python

Generating a Sample Dataset

Exploratory Data Analysis (EDA)

Data Visualization

Machine Learning Models

UAMS
University of Arkansas for Medical Sciences

Introduction to Python

What is Python?

- A **high-level**, **general-purpose** programming language.

UAMS
University of Arkansas for Medical Sciences

What is Python?

- A **high-level**, **general-purpose** programming language.
- Renowned for its **simple, clean syntax** that emphasizes readability.

UAMS
University of Arkansas for Medical Sciences

What is Python?

- A **high-level**, **general-purpose** programming language.
- Renowned for its **simple, clean syntax** that emphasizes readability.
- **Interpreted**, not compiled, which makes development faster and more interactive.

UAMS
University of Arkansas for Medical Sciences

What is Python?

- A **high-level**, **general-purpose** programming language.
- Renowned for its **simple, clean syntax** that emphasizes readability.
- **Interpreted**, not compiled, which makes development faster and more interactive.
- It's not just for data! Python is used for web development, automation, scientific computing, and much more.

UAMS
University of Arkansas for Medical Sciences

Why Python for Data Analysis?

Python has become the de-facto language for data science for several key reasons:

1. **Vast Ecosystem of Libraries:** This is its superpower. Specialized libraries provide powerful, pre-built functionality for nearly any data task.

UAMS
University of Arkansas for Medical Sciences

Why Python for Data Analysis?

Python has become the de-facto language for data science for several key reasons:

1. **Vast Ecosystem of Libraries:** This is its superpower. Specialized libraries provide powerful, pre-built functionality for nearly any data task.
2. **Ease of Learning:** Its straightforward syntax allows you to focus on solving problems rather than getting stuck on complex language rules.

UAMS
University of Arkansas for Medical Sciences

Why Python for Data Analysis?

Python has become the de-facto language for data science for several key reasons:

1. **Vast Ecosystem of Libraries:** This is its superpower. Specialized libraries provide powerful, pre-built functionality for nearly any data task.
2. **Ease of Learning:** Its straightforward syntax allows you to focus on solving problems rather than getting stuck on complex language rules.
3. **Large, Active Community:** If you have a problem, chances are someone has solved it and shared the solution online.

UAMS
University of Arkansas for Medical Sciences

Why Python for Data Analysis?

Python has become the de-facto language for data science for several key reasons:

1. **Vast Ecosystem of Libraries:** This is its superpower. Specialized libraries provide powerful, pre-built functionality for nearly any data task.
2. **Ease of Learning:** Its straightforward syntax allows you to focus on solving problems rather than getting stuck on complex language rules.
3. **Large, Active Community:** If you have a problem, chances are someone has solved it and shared the solution online.
4. **Integration and Versatility:** You can handle the entire data workflow—from data acquisition to machine learning—all in one language.

UAMS
University of Arkansas for Medical Sciences

Our Core Toolkit for Today

- **Pandas:** The essential library for data manipulation. It provides the **DataFrame**, a structure for handling tabular data.
  - *Our workhorse for importing, cleaning, filtering, and aggregating data.*

UAMS
University of Arkansas for Medical Sciences

Our Core Toolkit for Today

- **Pandas:** The essential library for data manipulation. It provides the **DataFrame**, a structure for handling tabular data.
  - *Our workhorse for importing, cleaning, filtering, and aggregating data.*
- **Matplotlib & Seaborn:** The primary libraries for creating a wide range of static and interactive visualizations.
  - *How we'll create histograms, bar charts, and scatter plots.*

UAMS
University of Arkansas for Medical Sciences

Our Core Toolkit for Today

- **Pandas:** The essential library for data manipulation. It provides the **DataFrame**, a structure for handling tabular data.
  - *Our workhorse for importing, cleaning, filtering, and aggregating data.*
- **Matplotlib & Seaborn:** The primary libraries for creating a wide range of static and interactive visualizations.
  - *How we'll create histograms, bar charts, and scatter plots.*
- **Scikit-learn & Statsmodels:** The go-to libraries for machine learning and statistical modeling.
  - *The tools we'll use to build our predictive and inferential models.*

UAMS
University of Arkansas for Medical Sciences

Generating a Sample Dataset

We start by generating a synthetic dataset.

```python
import pandas as pd
import numpy as np
import random

NUM_ROWS = 50000
sexes = ['Female', 'Male']
races = ['White', 'Black or African American', 'Asian', 'Other']

# Create a dictionary of data (simplified for slide)
data = {
    'Sex': [random.choice(sexes) for _ in range(NUM_ROWS)],
    'AgeOnIndexDate': np.random.randint(18, 85, size=NUM_ROWS),
    'FirstRace': [random.choice(races) for _ in range(NUM_ROWS)],
    'Depression': np.random.choice([0, 1], size=NUM_ROWS, p=[0.7, 0.3]),
    'ChronicPain': np.random.choice([0, 1], size=NUM_ROWS, p=[0.6, 0.4]),
    'ElixhauserScore': np.random.randint(0, 20, size=NUM_ROWS),
    'NumberOfEdVisits': np.random.randint(0, 15, size=NUM_ROWS)
}
df = pd.DataFrame(data)
df.to_csv('synthetic_patient_data.csv', index=False)
print("Data generated and saved.")
```

UAMS
University of Arkansas for Medical Sciences

Exploratory Data Analysis (EDA)

Load the data and get a high-level overview.

```
import pandas as pd
df = pd.read_csv('synthetic_patient_data.csv')

# Display first 5 rows
print("--- df.head() ---")
print(df.head())

# Display dataset info
print("\n--- df.info() ---")
df.info()
```

UAMS
University of Arkansas for Medical Sciences

Zoom in on specific subsets of the data.

```python
# Select a single column
ages = df['AgeOnIndexDate']

# Select multiple columns
demographics = df[['Sex', 'AgeOnIndexDate', 'FirstRace']]

# Filter rows based on a condition
# Patients with an Elixhauser Score greater than 15
high_risk_patients = df[df['ElixhauserScore'] > 15]

# Filter with multiple conditions (AND: &)
# Female patients with chronic pain
female_chronic_pain = df[(df['Sex'] == 'Female') & (df['ChronicPain'] == 1)]
```

UAMS
University of Arkansas for Medical Sciences

Calculate summary statistics for columns.

```
# Get summary statistics for all numerical columns
# Provides count, mean, std, min, max, and percentiles.
print(df.describe())

# Count occurrences in a categorical column
print("\n--- Value Counts for FirstRace ---")
print(df['FirstRace'].value_counts())
```

UAMS
University of Arkansas for Medical Sciences

Use groupby to split-apply-combine.

```
# Group by one column and calculate a single metric
# What is the average age for each sex?
avg_age_by_sex = df.groupby('Sex')['AgeOnIndexDate'].mean()
print(avg_age_by_sex)


# Group by a column and apply multiple aggregations
# Get stats for ElixhauserScore for each race
race_summary = df.groupby('FirstRace')['ElixhauserScore'].agg(
    ['mean', 'max', 'count']
)
print(race_summary)
```

UAMS
University of Arkansas for Medical Sciences

Data Visualization

Use a histogram to see the spread of a numerical variable.

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_theme(style="whitegrid")

# Histogram of patient ages
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='AgeOnIndexDate', bins=30, kde=True)
plt.title('Distribution of Patient Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

UAMS
University of Arkansas for Medical Sciences

Use a count plot to see the frequency of categories.

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_theme(style="whitegrid")

# Count plot of patients by race
plt.figure(figsize=(10, 6))
sns.countplot(data=df, y='FirstRace',
              order=df['FirstRace'].value_counts().index)
plt.title('Number of Patients by Race')
plt.xlabel('Count')
plt.ylabel('Race')
plt.show()
```

UAMS
University of Arkansas for Medical Sciences

Machine Learning Models

Convert categorical data to numbers and split into train/test sets.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Convert categorical variables into dummy/indicator variables
df_processed = pd.get_dummies(df, drop_first=True)

# Define Features (X) and Target (y)
y = df_processed['ReceivedMOUD']
X = df_processed.drop('ReceivedMOUD', axis=1)

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42, stratify=y)

# Scale numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

UAMS
University of Arkansas for Medical Sciences

Predicting a binary outcome: 'ReceivedMOUD'.

```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score

# Train the model
log_reg = LogisticRegression(random_state=42, max_iter=1000)
log_reg.fit(X_train, y_train)

# Make predictions and evaluate
y_pred = log_reg.predict(X_test)
print(f"Accuracy: {accuracy_score(y_test, y_pred):.4f}")
print(classification_report(y_test, y_pred))
```

UAMS
University of Arkansas for Medical Sciences

Using a more powerful model for prediction.

```python
from sklearn.ensemble import RandomForestClassifier

# Train the model
rf_clf = RandomForestClassifier(n_estimators=100, random_state=42, n_jobs=-1)
rf_clf.fit(X_train, y_train)

# Make predictions and evaluate
y_pred_rf = rf_clf.predict(X_test)
print(f"Accuracy: {accuracy_score(y_test, y_pred_rf):.4f}")
print(classification_report(y_test, y_pred_rf))
```

UAMS
University of Arkansas for Medical Sciences

Goal: Understand the effect of a predictor on a continuous outcome.

```python
import statsmodels.api as sm

# Use the original, unscaled dataframe for easier interpretation
# Y = continuous outcome, X = predictors
Y_lin = df['NumberOfEdvisitUniqueEncounter']
X_lin = df[['totalMorphineDose', 'AgeOnIndexDate']]

# We must add a constant (intercept) to our predictors
X_lin = sm.add_constant(X_lin)

# Fit the Ordinary Least Squares (OLS) model
model_lin = sm.OLS(Y_lin, X_lin).fit()

# Print the detailed summary
print(model_lin.summary())
```

UAMS
University of Arkansas for Medical Sciences

Goal: Model a count outcome, like number of ED visits.

```python
import statsmodels.api as sm

# Y = count outcome, X = predictors
Y_count = df['NumberOfEdvisitUniqueEncounter']
X_count = df[['ChronicPain', 'AgeOnIndexDate', 'Depression']]
X_count = sm.add_constant(X_count)

# Fit the Poisson model
# We use .GLM (Generalized Linear Model) with the Poisson family
poisson_model = sm.GLM(Y_count, X_count, family=sm.families.Poisson()).fit()

# Print the summary
print(poisson_model.summary())
```

UAMS
University of Arkansas for Medical Sciences

Goal: Model a count outcome with overdispersion.

```
import statsmodels.api as sm

# Use the same Y and X as the Poisson model
Y_count = df['NumberOfEdvisitUniqueEncounter']
X_count = df[['ChronicPain', 'AgeOnIndexDate', 'Depression']]
X_count = sm.add_constant(X_count)

# Fit the Negative Binomial model
# A good alternative to Poisson if the variance of the count is
# much larger than its mean (overdispersion).
neg_bin_model = sm.GLM(Y_count, X_count,
                       family=sm.families.NegativeBinomial()).fit()

# Print the summary
print(neg_bin_model.summary())
```

UAMS
University of Arkansas for Medical Sciences

Modeling a binary outcome for causal inference.

```python
import pandas as pd
import statsmodels.api as sm

# Define the dependent (Y) and independent (X) variables
Y = df['ReceivedMOUD'] # Binary outcome (0 or 1)
X = df[['ChronicPain', 'AgeOnIndexDate', 'Depression']]

# Add a constant (intercept) to the model
X = sm.add_constant(X)

# Instantiate and fit the Logit model
logit_model = sm.Logit(Y, X).fit()

# Print the detailed statistical summary
print(logit_model.summary())
```

UAMS
University of Arkansas for Medical Sciences

OLS vs Poisson vs Negative Binomial Regression

Feature	OLS	Poisson	Negative Binomial
Outcome Type	Continuous	Count	Count (overdispersed)
Data Example	Blood pressure, BMI	Number of ER visits	Number of ER visits
Assumes	Normality, constant variance	Mean = Variance	Variance Mean
Common Pitfall	Poor with skewed data	Poor fit with overdispersion	Better fit for overdispersed data
Interpretation	Change in mean outcome	Change in event rate (IRR)	Change in event rate (IRR)
Model Output	Linear coefficient (β)	Incidence Rate Ratio (IRR)	Incidence Rate Ratio (IRR)

UAMS
University of Arkansas for Medical Sciences

Thank You & Questions

Questions?

All codes and training materials will be found here:
https://github.com/Nahiyan140212/RTEC-Training

UAMS
University of Arkansas for Medical Sciences