Introduction to Programming with Python

Dr. Anatol Wegner

Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilians-Universität Würzburg
Würzburg, Germany

anatol.wegner@uni-wuerzburg.de

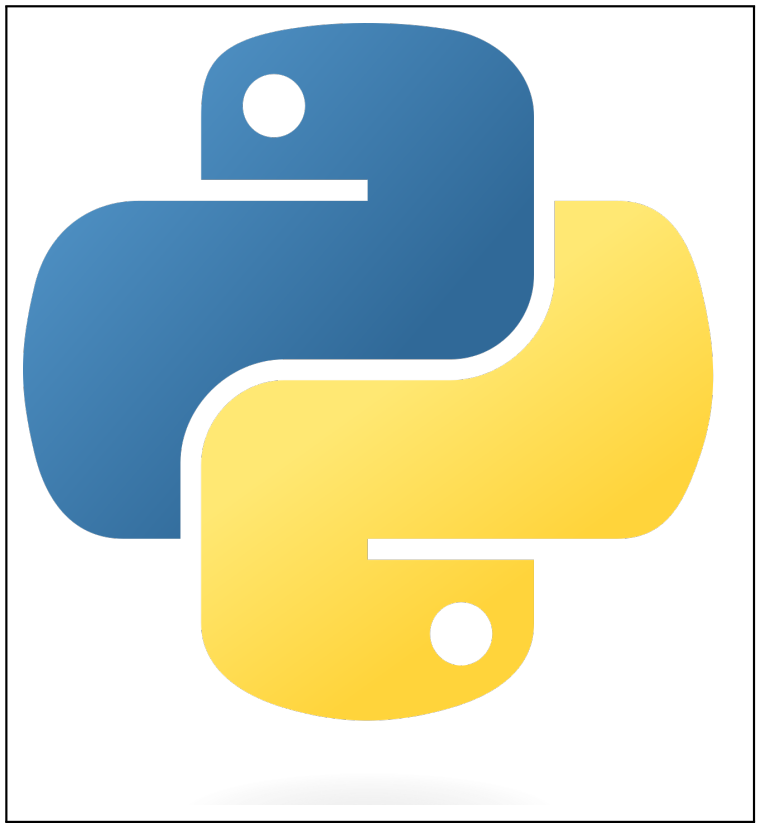
Lecture 06
Introduction to Machine Learning

December 06, 2024

Recap

Pandas: Data Analysis and Manipulation

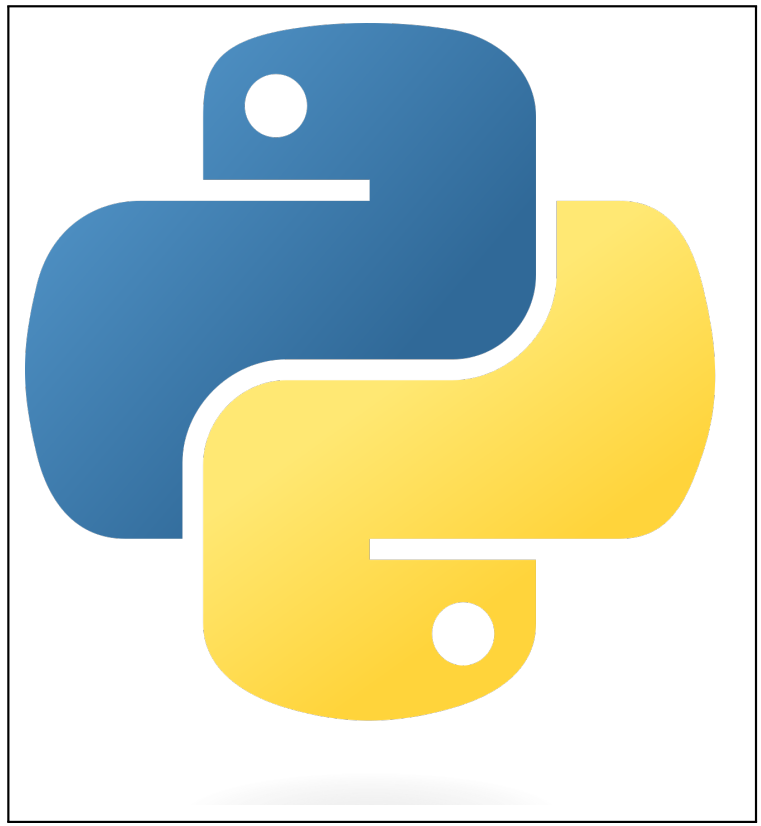
- Core Data Structures: `Series`, `DataFrame`.
- Data Import and Export: Functions like `read_csv()`, `to_csv()`.
- Data Exploration: `head()`, `info()`, `describe()`.
- Data Preprocessing: Handling missing data, filtering rows, and modifying columns.
- Grouping and Aggregation: Using `groupby()` for insights.
- Statistical Functions: `mean()`, `median()`, `std()`.



Recap

Seaborn: Data Visualization

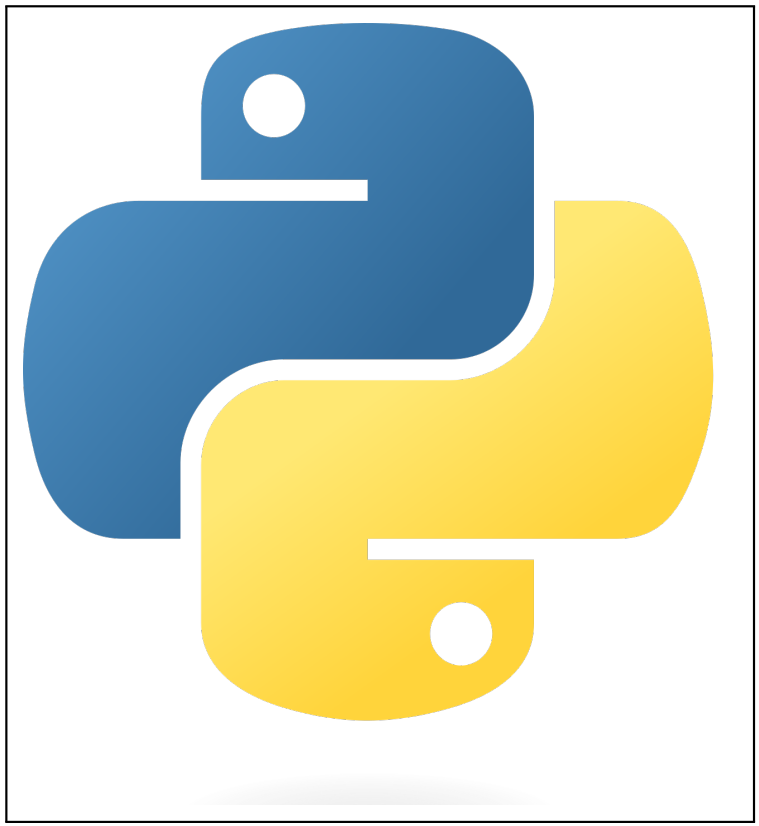
- High-Level Interface: Simplified plotting for statistical graphics.
- Common Plot Types: Scatter plots, bar plots, histograms, box plots.
- Customization: Themes, color palettes, and labels.
- Integration with Pandas: Directly using DataFrames for visualization.



Recap

Seaborn: Data Visualization

- High-Level Interface: Simplified plotting for statistical graphics.
- Common Plot Types: Scatter plots, bar plots, histograms, box plots.
- Customization: Themes, color palettes, and labels.
- Integration with Pandas: Directly using DataFrames for visualization.



Today: Machine Learning basics with scikit-learn

- ML workflow
- Supervised & unsupervised ML

What is Machine Learning?

Definition: Machine Learning (ML) is a branch of computer science that focuses on developing algorithms and statistical models capable of identifying patterns in data and making predictions or decisions based on data.

Key Idea: Instead of explicitly programming rules, ML systems use data to infer patterns and relationships.

Examples of ML applications:

- Text classification: Detecting spam emails.
- Geography: Grouping regions by climate data.
- Digital Humanities: Identifying sentiment in historical texts.

Key ML Paradigms

Supervised Learning:

- Learn a function from labeled data (features + target).
- Example: Predicting house prices from size, location, etc.

Unsupervised Learning:

- Discover patterns in data without predefined labels.
- Example: Grouping customers based on purchase history.

Others (not covered today):

- Reinforcement Learning: Learning by interacting with an environment.

The ML Workflow

Step-by-step process:

1. **Problem Definition:** What are we predicting or discovering?
2. **Data Preparation:** Cleaning, transforming, and exploring data (pandas, seaborn).
3. **Model Selection:** Choose a suitable algorithm (e.g., regression, clustering).
4. **Training and Evaluation:** Fit the model and assess performance.
5. **Prediction:** Use the trained model on new data.

Step 2: `pandas`, `seaborn`

Steps 3–5: `Scikit-learn`

Introducing Scikit-learn

What is Scikit-learn? (aka sklearn)

- A widely-used Python library for machine learning.
- Provides tools for supervised and unsupervised learning, model evaluation, and preprocessing.
- Built on top of NumPy, SciPy, and matplotlib for high performance.

Why use Scikit-learn?

- Easy integration with pandas for data manipulation.
- Works seamlessly with seaborn for data visualization.
- Unified interface for training, testing, and evaluating ML models.

Documentation: `https://scikit-learn.org/stable/`

Introducing Supervised Learning

What is Supervised Learning?

- A machine learning paradigm where the model learns from labeled data (features and their corresponding targets).
- Goal: Predict outcomes (labels) for unseen data based on training data.

Example: MNIST Digit Classification

- Dataset: Handwritten digits (0–9) represented as 28x28 pixel images (flattened into 784 features).
- Task: Predict the correct digit based on the pixel values.

Introducing Supervised Learning

What is Supervised Learning?

- A machine learning paradigm where the model learns from labeled data (features and their corresponding targets).
- Goal: Predict outcomes (labels) for unseen data based on training data.

Example: MNIST Digit Classification

- Dataset: Handwritten digits (0–9) represented as 28x28 pixel images (flattened into 784 features).
- Task: Predict the correct digit based on the pixel values.

K Nearest Neighbour (KNN)

- KNN is a simple, yet powerful algorithm for classification.
- It works by finding the 'k' closest data points to a new observation and assigning the most common label among them.
- KNN is often a good starting point for classification problems.

Example: KNN Classification on MNIST

Step 1: Import Libraries

- Import necessary libraries for loading data, preprocessing, and modeling.

```
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
```

Example: KNN Classification on MNIST

Step 2: Load the MNIST Dataset

- MNIST dataset contains 70,000 28x28 grayscale images of handwritten digits.

```
mnist = fetch_openml('mnist_784', version=1)
X, y = mnist["data"], mnist["target"]
```

Example: KNN Classification on MNIST

Step 3: Split the Data into Training and Testing Sets

- Split the data into 80% training and 20% testing.

```
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2)
```

Example: KNN Classification on MNIST

Step 4: Initialize the KNN Classifier and Train the Model

- Use KNN with 3 neighbors (k=3) to classify the digits.

```
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
```

Example: KNN Classification on MNIST

Step 5: Evaluate the Model

- Predict labels for the test set and calculate the accuracy.

```
y_pred = knn.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

- sk-learn contains a variety of models for supervised classification.
- all such models can used following the same general `.fit()` and `.predict()` routine.

Summary of Supervised Machine Learning

- **Supervised Learning Overview**:
  - The model is trained on input-output pairs, where the output is known (labeled data).
- **Key methods in sk-learn**:
  - `train_test_split(X, y)` is used to split the data into training and testing sets.
  - `model.fit(X_train, y_train)` is used to train the model on the training data.
  - `model.predict(X_test)` is used to make predictions on new, unseen data using the trained model.

Summary of Supervised Machine Learning

- **Supervised Learning Overview**:
  - The model is trained on input-output pairs, where the output is known (labeled data).
- **Key methods in sk-learn**:
  - `train_test_split(X, y)` is used to split the data into training and testing sets.
  - `model.fit(X_train, y_train)` is used to train the model on the training data.
  - `model.predict(X_test)` is used to make predictions on new, unseen data using the trained model.

Practice Session 1

- Supervised classification with scikit-learn.

`https://gitlab2.informatik.uni-wuerzburg.de/ml4nets_notebooks/2024_wise_infhaf_notebooks/-/blob/main/PythonIntroNotebooks/Lecture_06.ipynb`

Introduction to Unsupervised Learning

In unsupervised learning the model is trained using data that has no labeled outcomes or target variables. The goal is to identify underlying patterns, structures, or relationships within the data.

- **No Labeled Data**: Unlike supervised learning, where the model learns from labeled input-output pairs, unsupervised learning works with unlabeled data, aiming to find hidden structures or representations.
- **Applications**:
  - **Clustering**: Grouping similar data points together, for example, customer segmentation or document clustering.
  - **Dimensionality Reduction**: Reducing the number of features while retaining important information
  - **Anomaly Detection**: Identifying unusual or rare data points
- **Key Challenge**: evaluating model performance without explicit feedback. The interpretation of the results often requires domain knowledge.

Common Unsupervised Learning Algorithms

Unsupervised learning methods include various approaches such as clustering, dimensionality reduction, and density estimation. Some popular algorithms are:

- **K-Means Clustering**: A method to partition data into K clusters based on similarity.
- **DBSCAN (Density-Based Spatial Clustering)**: Identifies clusters of varying shapes based on density.
- **Principal Component Analysis (PCA)**: A technique for reducing the number of features while retaining variance.
- **t-SNE (t-Distributed Stochastic Neighbor Embedding)**: A method for visualizing high-dimensional data in 2D or 3D.

K-Means Clustering

K-Means is one of the most widely used clustering algorithms. It aims to partition the dataset into K clusters, minimizing the variance within each cluster.

- The algorithm assigns each data point to the nearest centroid.
- The centroids are updated iteratively to minimize the sum of squared distances between data points and their corresponding centroids.
- Suitable for problems with well-defined 'spherical' clusters.

Implementing K-Means in Scikit-learn with the MNIST Dataset

In this example, we aim to cluster the MNIST dataset into 10 groups, corresponding to the 10 possible digits (0-9).

```
from sklearn.cluster import KMeans

# Load the MNIST dataset
mnist = fetch_openml('mnist_784', version=1)
X = mnist["data"].values  # Image features

# Apply K-Means clustering with 10 clusters (for digits 0-9)
kmeans = KMeans(n_clusters=10, random_state=42)
kmeans.fit(X)
```

DBSCAN Clustering with Scikit-learn

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering algorithm that groups points based on density. It can discover clusters of arbitrary shape and is robust to outliers.

- **Parameters**:
  - `eps`: Maximum distance between two samples to be considered as in the same neighborhood.
  - `min_samples`: Minimum number of samples in a neighborhood to form a core point.
- **Advantages**:
  - Identifies clusters of arbitrary shape.
  - Handles noise (outliers) naturally.
  - Does not require the number of clusters to be specified.

```
from sklearn.cluster import DBSCAN

# Apply DBSCAN clustering
dbscan = DBSCAN(eps=0.2, min_samples=5)
dbscan.fit(X)
```

Conclusion

Unsupervised learning provides powerful tools for exploring and understanding data. Key methods include:

- **Clustering**: Grouping similar data points (e.g., K-Means, DBSCAN).
- **Dimensionality Reduction**: Reducing feature space while retaining important information (e.g., PCA, t-SNE).
- **Anomaly Detection**: Identifying outliers and unusual data points.

Conclusion

Unsupervised learning provides powerful tools for exploring and understanding data. Key methods include:

- **Clustering**: Grouping similar data points (e.g., K-Means, DBSCAN).
- **Dimensionality Reduction**: Reducing feature space while retaining important information (e.g., PCA, t-SNE).
- **Anomaly Detection**: Identifying outliers and unusual data points.

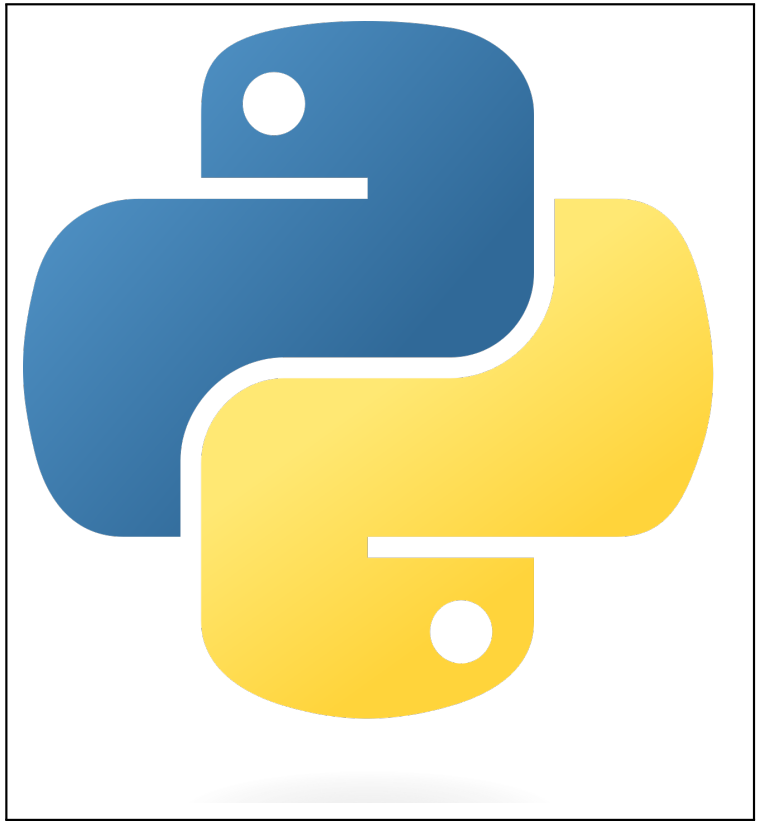
Practice Session 2

- Clustering MNIST with sk-learn.

`https://gitlab2.informatik.uni-wuerzburg.de/ml4nets_notebooks/2024_wise_infhaf_notebooks/-/blob/main/PythonIntroNotebooks/Lecture_06.ipynb`

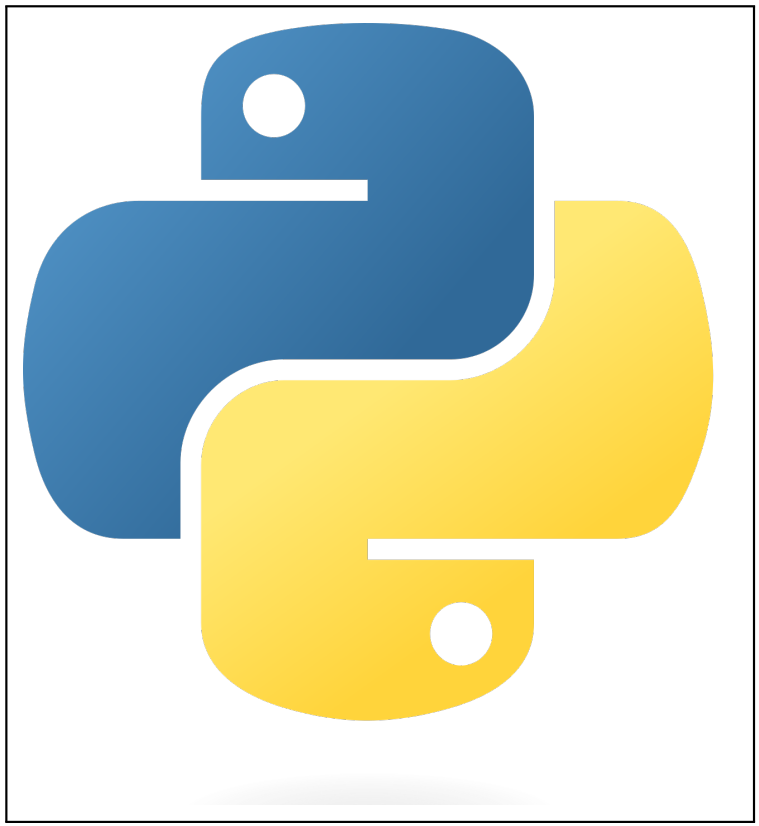
In summary

- **Supervised Learning**:
  - **Definition**: Models learn from labeled data to predict outputs for new inputs.
  - **Key Concepts**:
    - Classification (e.g., MNIST digit classification).
    - Algorithms: `KNeighborsClassifier`, SVC.
    - Train-Test Split: Preparing data for training and evaluation.



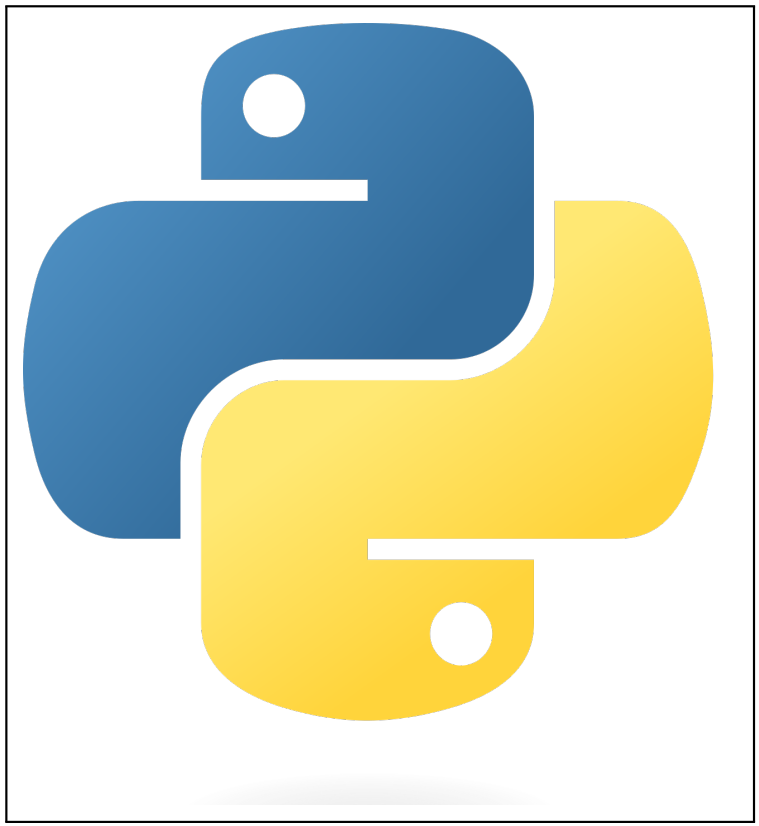
In summary

- **Unsupervised Learning**:
  - **Definition**: Models learn patterns and structures from unlabeled data.
  - **Key Concepts**:
    - Clustering (e.g., DBSCAN, K-Means on MNIST).



In summary

- **Scikit-learn Basics**:
  - Unified interface for training (`fit`) and predictions (`predict`).
  - Tools for preprocessing, model evaluation, and visualization.



Exercise Session

- Supervised and unsupervised ML with sklearn

`https://gitlab2.informatik.uni-wuerzburg.de/ml4nets_notebooks/2024_wise_infhaf_notebooks/-/blob/main/PythonIntroNotebooks/Exercise_L06.ipynb`

Self-Study Questions:

1. What is the difference between unsupervised learning and supervised learning?
2. What does the `train_test_split` method in sklearn do?
3. Can you briefly describe the k-Nearest Neighbors (KNN) algorithm?
4. How can you evaluate the performance of a supervised classification algorithm ?
5. How does the K-means algorithm determine clusters ?
6. How can clustering results be evaluated without labeled data?
7. What is the role of the `fit` and `predict` methods in sklearn?
8. What steps would you take to visualize and interpret the results of an unsupervised learning algorithm?