

<i>Course:</i>	Maschinelles Lernen und künstliche Intelligenz/ Machine Learning and Artificial Intelligence		
<i>(Group) Member:</i>	<i>Nr.</i>	<i>Name</i>	<i>Matrikel</i>
	1	Miguel Angel Lopez Mejia	5197617
	2		
	3		

Task Reporting Template

Please read the given Task carefully. You can find all necessary information in Moodle.

1. Definition

1.1. Introduction and Explanations

Task 4: Simple statistical classifier in python for Iris

Your botanist friend needs your help to identify the species of given feature values.

You know that statistical classification can be used to classify the type of iris flower from a given feature set (from the iris dataset). You decide to come to the rescue and create an amazing Python application - you name the application **"Iris_Species.py"**.

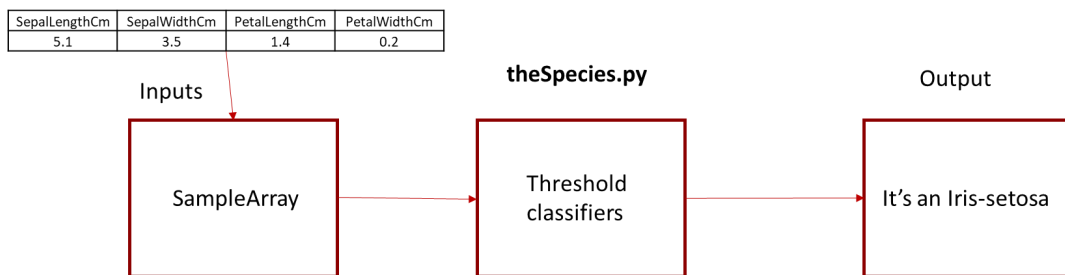
The application works by asking the botanist via terminal the feature values of the specific sample they want to classify. The user should be able to classify as many samples as they need (in a loop, controlled by "yes" or "no" user-input) before being exhausted by the sheer brilliance of your application and deciding to stop for the day by simply typing the letter "N" for No. |

Hints:

Step 1: Use your investigation outputs from task 3 to create thresholds as classifiers in pseudo code.

Step 2: Use the pseudo code to help you in creating your python application – Iris_Species.py

The application needs to classify the type of flower from the given feature set, as shown as an example below. Please test your application with the given feature-sets of iris-dataset and determine the accuracy (this is the error rate – so in words x% of tested feature sets returned a correct result of classification/ label).



You find the development so easy, you don't need any libraries, but if you must, then you need to justify why you are using the selected library.

1.2. Allowed Tools

- The script is going to be executed using **Python version 3.13.2**
- **Visual Studio Code** is going to be used as IDE (Integrated Development Environment)
- **Libraries:** pandas, numpy, logging

2. Your Preparation/ Concept

2.1. Structure your Problem

- **Task 4. Simple statistical classifier in python for Iris**
 - Create the pseudocode
 - Create the function that will classify the flowers introduced by the user
 - Test the accuracy of the function by using the Iris dataset
 - Test different approaches to decide which algorithm has a better accuracy

Flows:

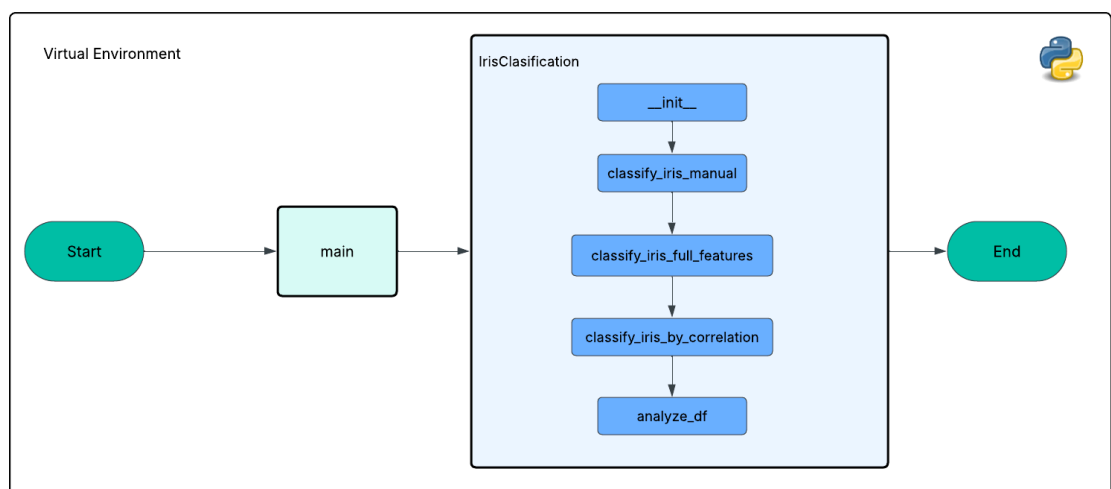


Figure 1: Iris analysis diagram

Libraries:

- **matplotlib** is a comprehensive library for creating static, animated, and interactive visualizations.
- **logging**: this module defines functions and classes which implement a flexible event logging system for applications and libraries.
- **numpy** is a Python library providing a multidimensional array object, derived objects like masked arrays and matrices, and tools for fast array operations, including math, logic, shape manipulation, sorting, I/O, Fourier transforms, linear algebra, statistics, and random simulation.
- **Seaborn** is a Python data visualization library built on top of Matplotlib. It focuses on making it easier to create attractive and informative statistical graphics. Seaborn simplifies the process of creating complex visualizations with just a few lines of code and provides a high-level interface for drawing various types of statistical plots.

```
# region Libraries  
import matplotlib.pyplot as plt  
import seaborn  
import logging  
import numpy as np  
from sklearn.datasets import load_iris  
# endregion
```

2.2. Your Implementation

Virtual Environment

- In order to run the script, a virtual environment must be created, open a new terminal and create it as it is shown:

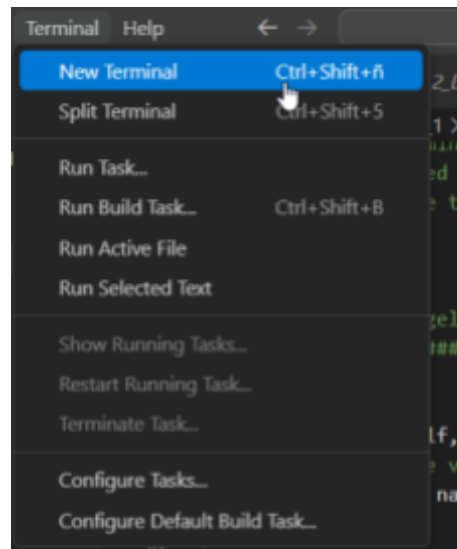


Figure 2: New terminal

- Execute the following command in the console in order to create a new virtual environment: ***python -m venv venv*** once the virtual environment is created, it will be displayed in the explorer section:

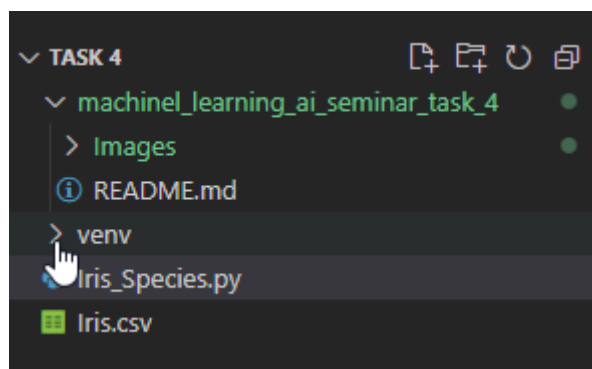


Figure 3: Virtual environment

- In order to activate the virtual environment, execute this script in the console: ***venv/scripts/activate .***

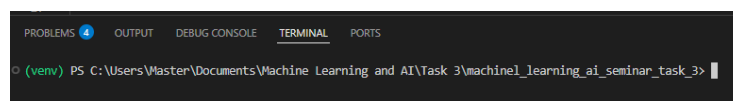


Figure 4: activation of venv

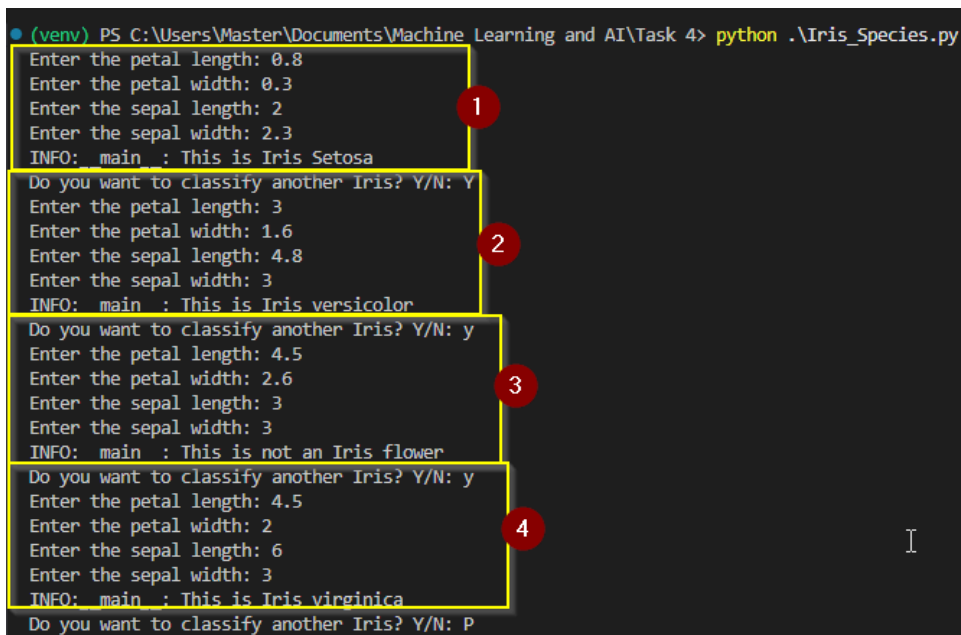
Task 4.1 - Simple statistical classifier in python for Iris

In order to execute the script, run this command in the console: `python .\Iris_Species.py`,

Step 1: I created a pseudocode that will help me to elaborate the script:

```
def pseudocode(self):  
  
    # In a while loop that depends on an input (if the user introduces a N or n the loop will finish), ask the  
    user to introduce the values of the features  
    # Create a list that will contain the feature values  
    # Create a method that receives the 4 features within a list  
    # Extract them  
    # Compare the values of the petal and sepal, depending on the range of values, return ("This is Iris  
    Setosa", "This is Iris versicolor",  
    # "This is Iris virginica")  
    # If the values don't match any range, return a "This is not an Iris flower".  
  
    pass
```

Step 2: The script will ask to the user to introduce the values of the features and it will classify them, after that, it will ask the user if he/she wants to classify another flower:



```
(venv) PS C:\Users\Master\Documents\Machine Learning and AI\Task 4> python .\Iris_Species.py  
Enter the petal length: 0.8  
Enter the petal width: 0.3  
Enter the sepal length: 2  
Enter the sepal width: 2.3  
INFO: main : This is Iris Setosa  
Do you want to classify another Iris? Y/N: Y  
Enter the petal length: 3  
Enter the petal width: 1.6  
Enter the sepal length: 4.8  
Enter the sepal width: 3  
INFO: main : This is Iris versicolor  
Do you want to classify another Iris? Y/N: y  
Enter the petal length: 4.5  
Enter the petal width: 2.6  
Enter the sepal length: 3  
Enter the sepal width: 3  
INFO: main : This is not an Iris flower  
Do you want to classify another Iris? Y/N: y  
Enter the petal length: 4.5  
Enter the petal width: 2  
Enter the sepal length: 6  
Enter the sepal width: 3  
INFO: main : This is Iris virginica  
Do you want to classify another Iris? Y/N: P
```

Figure 5: Results of the script

The script also verifies that the response from the user is correct, so, if a not valid input is entered it ask for the correct value:

```
Do you want to classify another Iris? Y/N: P
Please enter an acceptable response, Do want to classify another Iris? Y/N: Y
Enter the petal length: 4.5
Enter the petal width: 2
Enter the sepal length: 6
Enter the sepal width: 3
INFO:__main__: This is Iris virginica
Do you want to classify another Iris? Y/N: N
(venv) PS C:\Users\Master\Documents\Machine Learning and AI\Task 4> |
```

Figure 6: Script evaluates the input from the user

Finally, I validated the algorithm with all the features using the Iris dataset and I created a second algorithm with only the features that have more correlation and that have more independence between them. Here are the results.

All features:

- The accuracy is 96.67 %

Petal features:

- The accuracy is 96.67 %

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Master\Documents\Machine Learning and AI\Task 4> venv/scripts/activate
(venv) PS C:\Users\Master\Documents\Machine Learning and AI\Task 4> python .\Iris_Species.py
INFO:__main__: This is the accuracy of the model using all the features: 96.67 %
INFO:__main__: This is the accuracy of the model using only petal width and petal length: 96.67 %
(venv) PS C:\Users\Master\Documents\Machine Learning and AI\Task 4> |
```

Figure 7: Accuracy of the algorithms

Conclusion

By using the results obtained in the previous tasks I was able to identify the features that have more independence between them. These features are the Petal Width and Petal Length, by running the script using the Iris dataset, I concluded that by only using 2 of the 4 features, the accuracy was the same, which means that the computation will decrease and the performance of the script will be better when using huge datasets.

As an extra, I used the library pandas, seaborn and matplotlib to create the correlation of the features, where we can appreciate that the Petal feature is the nearest to 1.

```
# Get correlation of the features
matrix_correlacion = df[['SepalLengthCm', 'SepalWidthCm',
                          'PetalLengthCm', 'PetalWidthCm']].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(matrix_correlacion, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Correlation Matrix')
plt.show()
# Extract the feature values
x_train = df.drop(columns=['Id', 'Species']).values
map_species_target = {
    'Iris-setosa': 0,
    'Iris-versicolor': 1,
    'Iris-virginica': 2
}
# Map each specie to a number
y_train = df['Species'].map(map_species_target).values
```

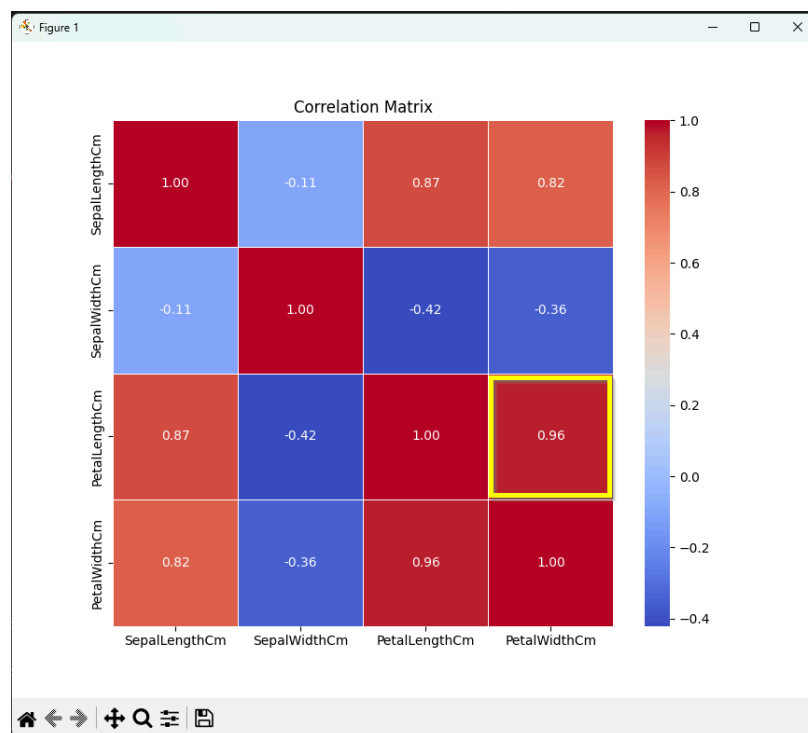


Figure 8: Correlation of the features

Github Repository

[MiguelAnhalt/machine learning ai seminar task 2](#)

References, extra reading links, and resources

- Dua, D., & Graff, C. (2017). *UCI Machine Learning Repository* <http://archive.ics.uci.edu/ml>. Irvine, CA: University of California, School of Information and Computer Science.
- NIST/SEMATECH. (n.d.). *Box Plot*. In *Engineering Statistics Handbook*. Retrieved May 2, 2025, from <https://www.itl.nist.gov/div898/handbook/eda/section3/boxplot.htm>
- GeeksforGeeks. (2023, November 20). *Iris Dataset*. <https://www.geeksforgeeks.org/iris-dataset/>
- W3Schools. (n.d.). *Python Classes and Objects*. https://www.w3schools.com/python/python_classes.asp