

**TRIBHUVAN UNIVERSITY**

**National College of Computer Studies (NCCS­­­)**

Paknajol, Kathmandu, Nepal

Project Report

Python Project – Machine Learning

Iris Flower Classification

Submitted By: Submitted To:

Nicky Maharjan Mausam Rajbanshi

BIM 5th semester

Section: B

Roll no.: 10

# Acknowledgment

In the pursuit of data classification for the Iris flower dataset using Python, it is important to express my gratitude to the various individuals and resources who have aided in the successful completion of this endeavor. First and foremost, I want to thank Ronald A. Fisher, whose 1936 creation of the Iris dataset laid the groundwork for my research into machine learning and data analysis. I also want to thank the scikit-learn library's developers and contributors, whose dedication to open-source software has provided critical tools for this project.

Furthermore, the collaborative efforts of the Python community, as well as the wealth of educational resources available online, have been invaluable in guiding my efforts. I am grateful for the guidance and support provided by academic and professional mentor, Mausam Rajbanshi. Finally, throughout this journey, my colleagues and peers have provided invaluable assistance, feedback, and collaborative insights. This recognition is a tribute to the collaborative spirit and collective effort that have enriched my understanding and application of data classification techniques.

Table of Contents

[Acknowledgment ii](#_Toc144729743)

[Introduction to Machine Learning 1](#_Toc144729744)

[Machine Learning in Python 2](#_Toc144729745)

[Background 3](#_Toc144729746)

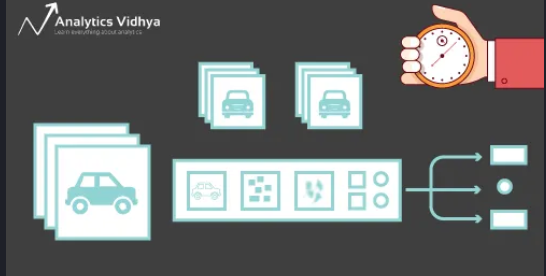
[Objective 5](#_Toc144729747)

[Implementation 6](#_Toc144729748)

[Conclusion 17](#_Toc144729749)

# Introduction to Machine Learning

Machine learning is a subfield of AI that focuses on the development of algorithms and statistical models that allow computer systems to learn and improve their performance on a specific task without being explicitly programmed. It boils down to teaching machines to learn from data and then make predictions or decisions based on that learning.





Here are the key machine learning components and concepts:

* Data: The foundation of machine learning is data. Data is used to teach machine learning models patterns and relationships. This data can be structured (for example, tables and databases) or unstructured (for example, text, images, and audio). The quality and quantity of data frequently have a significant impact on machine learning model performance.
* Training: A machine learning model is trained by exposing it to a dataset containing input data (features) and corresponding target values or labels. By adjusting its internal parameters, the model learns to recognize patterns or relationships in data.
* Features: Features are the variables or attributes that are used to describe the data. The process of selecting, transforming, or creating meaningful features that help the model learn and make accurate predictions is known as feature engineering.
* Model: A machine learning model is a mathematical representation of data patterns or relationships. For various tasks, various models are used, such as linear regression for regression problems, decision trees for classification, and neural networks for complex tasks such as image recognition.
* Algorithm: Machine learning algorithms are mathematical and computational techniques that are used to train models. These algorithms differ according to the type of learning (supervised, unsupervised, or reinforcement) and the task at hand.
* Evaluation: Depending on the task, machine learning models are evaluated using various metrics and techniques such as accuracy, precision, recall, F1-score, or mean squared error.
* Deployment: Machine learning models can be deployed in real-world applications after training and evaluation to make predictions or automate decision-making processes.
* Continuous Learning: Machine learning models can be updated and improved as new data becomes available. This is known as online learning or retraining, and it is critical for maintaining model accuracy over time.

Machine learning is widely used in a variety of domains, including healthcare (diagnosis and prognosis), finance (fraud detection and stock prediction), natural language processing (language translation and sentiment analysis), image recognition (object detection and facial recognition), and many others. It continues to play an important role in advancing technology and solving complex problems.

## Machine Learning in Python

Python is a popular data science programming language due to its simplicity, versatility, and a rich ecosystem of libraries and tools designed specifically for data analysis and machine learning. Here's a quick rundown of how Python is used in data science:

* Data Manipulation and Analysis:
* NumPy: Numpy supports arrays and matrices, making it simple to perform numerical operations on large datasets.
* Pandas: Pandas is a powerful data manipulation and analysis library. It provides data structures such as DataFrames for dealing with structured data.
* Data Visualization:
* Matplotlib: Matplotlib is a popular library for creating static, animated, and interactive plots and charts.
* Seaborn: Seaborn is based on Matplotlib and provides a higher-level interface for creating visually appealing statistical graphics.
* Machine Learning:
* scikit-learn: scikit-learn is a large machine learning library that includes tools for classification, regression, clustering, and model selection.

Python's rich library ecosystem, combined with its readability and ease of use, makes it an excellent choice for data scientists looking to effectively explore, analyze, and model data. Depending on the task, you can use a variety of libraries and tools to achieve your data science objectives.

# Background

Data classification is a fundamental task in machine learning that involves training a model to classify data into predefined classes or categories. The Iris flower dataset, first introduced in 1936 by British biologist and statistician Ronald A. This dataset includes measurements of three Iris flower species: Iris setosa, Iris virginica, and Iris versicolor. Four distinct features are measured in centimeters for each flower sample: sepal length, sepal width, petal length, and petal width.

The primary goal of this data classification task is to construct a machine learning model capable of accurately classifying Iris flowers into their respective species based on these four attributes. To accomplish this, we use Python, a popular programming language for data analysis and machine learning, and a variety of libraries and techniques to handle the dataset, build a classification model, and evaluate its performance.

The following are the general steps involved in this data classification task:

* Data Collection: To begin, we obtain the Iris flower dataset, which is easily accessible in Python via the scikit-learn library. This dataset will be the basis for training and testing our classification model.
* Data Preparation: To facilitate data manipulation, the dataset is organized into a structured format, typically a DataFrame. Furthermore, the data is divided into two subsets: one for model training and one for model evaluation. To ensure consistent scaling of features, data preprocessing steps such as standardization may be used.
* Algorithm Selection: We select an appropriate machine learning algorithm for this classification task. In this example, we use the k-nearest neighbors (KNN) algorithm, which is a straightforward yet effective method for classification tasks. The algorithm chosen is determined by the nature of the data and the requirements of the problem.
* Model Training: Using the training dataset, the selected algorithm is trained. The model learns to recognize patterns and relationships in the data during this phase, allowing it to make predictions.
* Model Evaluation: The model is evaluated using the testing dataset after it has been trained. To evaluate the model's performance, metrics such as accuracy, precision, recall, and the F1-score are calculated. A confusion matrix is also generated to visualize the model's classification results.
* Interpretation and Reporting: Model evaluation results are interpreted and reported on to determine the model's accuracy and effectiveness in classifying Iris flowers. These findings are then documented, most commonly in the form of a classification report and a confusion matrix.

This basic background gives an overview of the steps required to perform data classification on the Iris flower dataset using Python. The process is invaluable for understanding the principles of data classification and model evaluation and serves as a foundational example for machine learning practitioners. Further investigation and experimentation with various algorithms and techniques can result in improved classification performance and insights.

# Objective

The goal of this report is to use Python to perform data classification on the Iris flower dataset. The purpose of this report is to introduce the Iris dataset, which consists of measurements from three distinct Iris flower species, as well as to detail the data preprocessing process. We'll go over the reasoning behind data cleaning, dealing with missing values, and standardizing features. The selection of a classification algorithm, specifically the k-nearest neighbors (KNN) algorithm, will be justified, and the model's training will be described. We will assess the model's performance using a variety of metrics, such as accuracy, precision provide a detailed interpretation of the results. To improve result visualization, visual aids such as a confusion matrix will be included.

In addition, we will discuss the practical implications of accurate species classification and suggest potential areas for further investigation and improvement. This report will conclude with a summary of key findings and recommendations for using the classification model effectively in practical applications.

# Implementation

In this section, we look at how to use sepal and petal measurements to classify Iris flowers into their respective species—specifically, Iris setosa, Iris virginica, and Iris versicolor. To carry out the classification task, we use Python, a versatile programming language for data analysis and machine learning. The following steps outline the key implementation elements:

1. Preparation of the Dataset:

The Iris flower dataset is loaded to begin the implementation. This dataset is easily accessible in Python via the scikit-learn library, which is a powerful machine learning tool. For each Iris flower sample, the dataset includes measurements of sepal length, sepal width, petal length, and petal width, as well as the corresponding target labels indicating the species.

2. Data Division:

We divide the dataset into two subsets to facilitate model training and evaluation: a training set and a testing set. In this implementation, 70% of the data is allocated to the training set, with the remaining 30% used to test the model's performance. This division ensures that the model learns from a subset of the data and is tested on an independent dataset to determine its ability to generalize.

3. Standardization of Features:

While it is not always required, standardizing the features is a common practice to ensure that all attributes have the same scale. We use scikit-learn's StandardScaler to standardize the feature values, which can be especially useful when working with certain classification algorithms.

4. Choosing an Algorithm:

We use the Logistic Regression, k-nearest neighbors (KNN) , decision tree classifier algorithm for this classification task.

5. Model Education:

The training set is used to train the chosen classifiers. During this stage, the model learns to recognize patterns and relationships in data, allowing it to make predictions about Iris flower species.

6. Model Assessment:

Following model training, we assess its performance with a variety of classification metrics. Accuracy, precision, recall are all important metrics. These metrics reveal how well the model distinguishes between the various Iris flower species.

7. Visualization of Results:

We include visual aids such as a confusion matrix to improve the presentation of results. A confusion matrix depicts the number of correct and incorrect classifications for each species, providing a clear visualization of the model's classification outcomes.

These steps are encapsulated in the implementation code, which can be found above. It not only demonstrates the technical execution of the classification task, but it also serves as the foundation for further analysis and interpretation of the results.

The codes used for the projects are given below:

#Import modules

import pandas as pd

import numpy as np

import os

import matplotlib.pyplot as plt

import seaborn as sns

#Read the data set

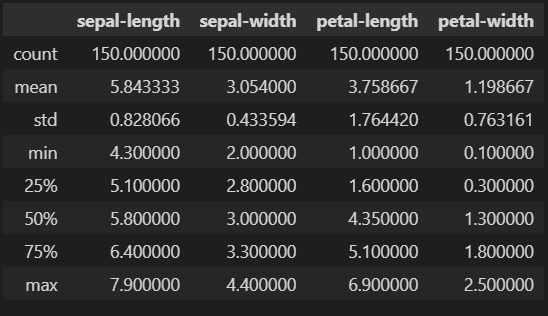
data=pd.read\_csv("D:\python\PythonSlides\iris-data.csv")

data.head()



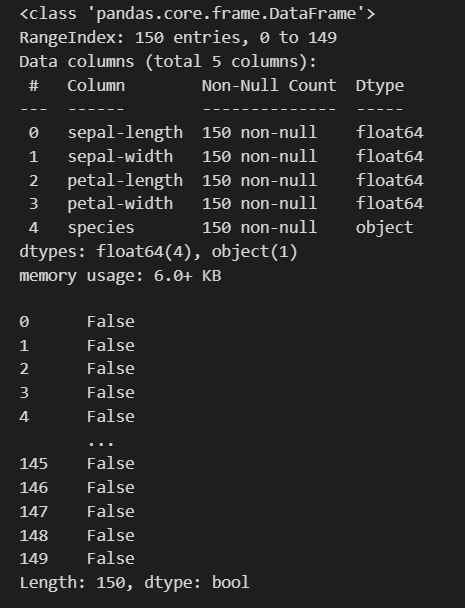
#Describing data set

data.describe()



data.info()

data.duplicated()



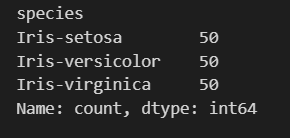
#Identify unique column

data['species'].unique()



#To display no. of samples on each class

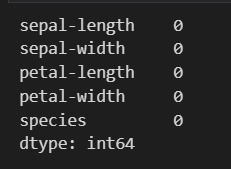
data['species'].value\_counts()



#Check for null values

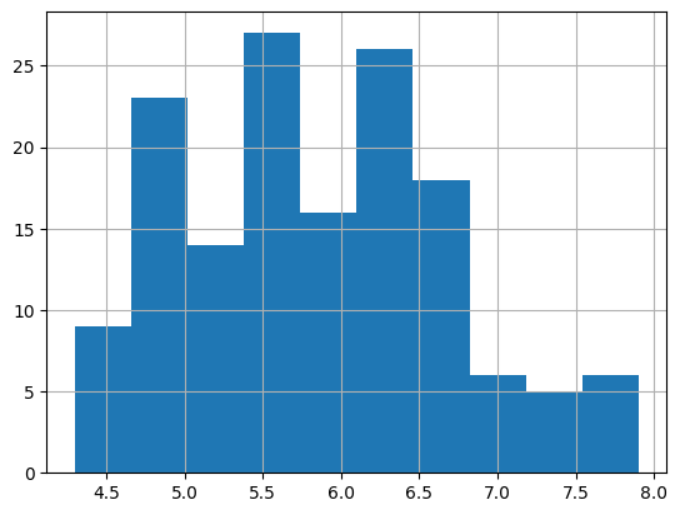
data.isnull()

data.isnull().sum()

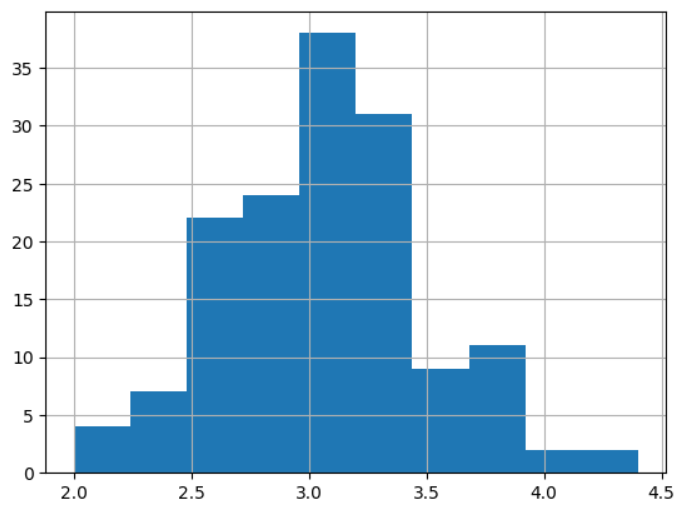


#Histogram

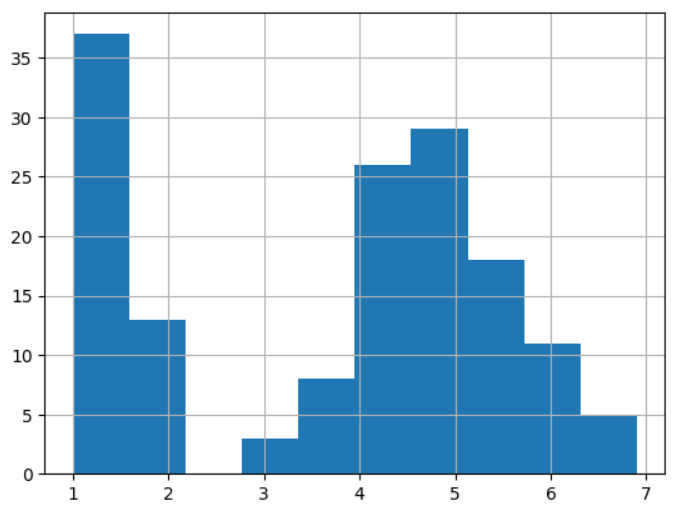
data['sepal-length'].hist()



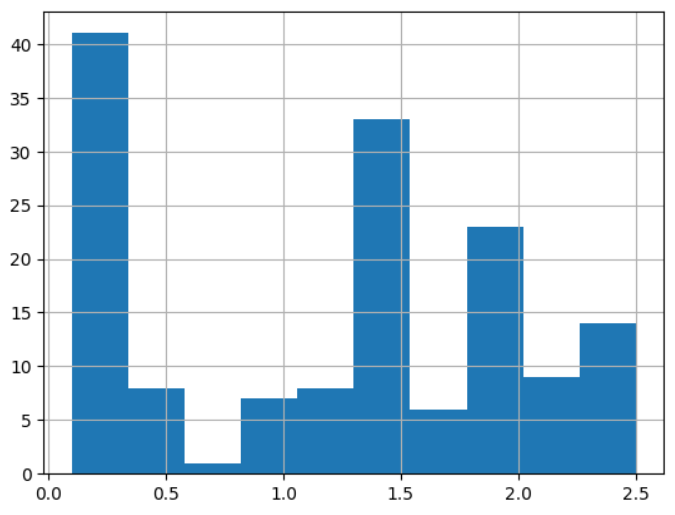
data['sepal-width'].hist()



data['petal-length'].hist()



data['petal-width'].hist()



#Scatterplot

colors=['red','orange','blue']

species=['Iris-virginica','Iris-versicolor','Iris-setosa']

for i in range(3):

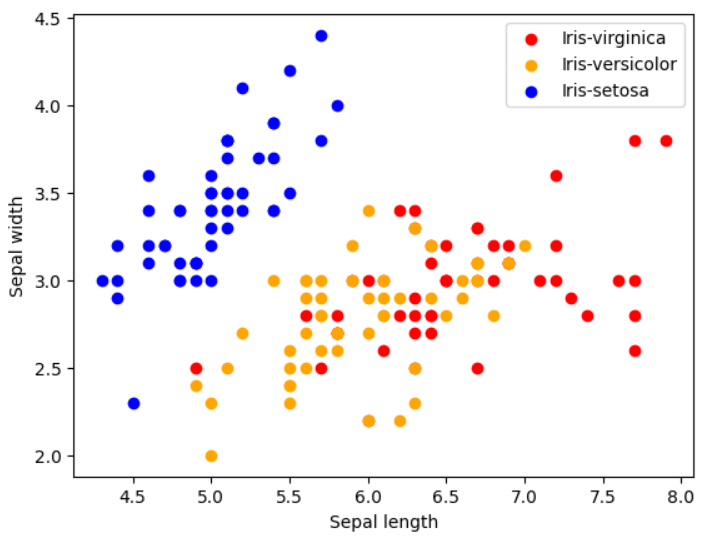
    x = data[data['species'] == species[i]]

    plt.scatter(x['sepal-length'], x['sepal-width'], colors[i],label=species[i])

    plt.xlabel("Sepal length")

    plt.ylabel("Sepal width")

    plt.legend()



for i in range(3):

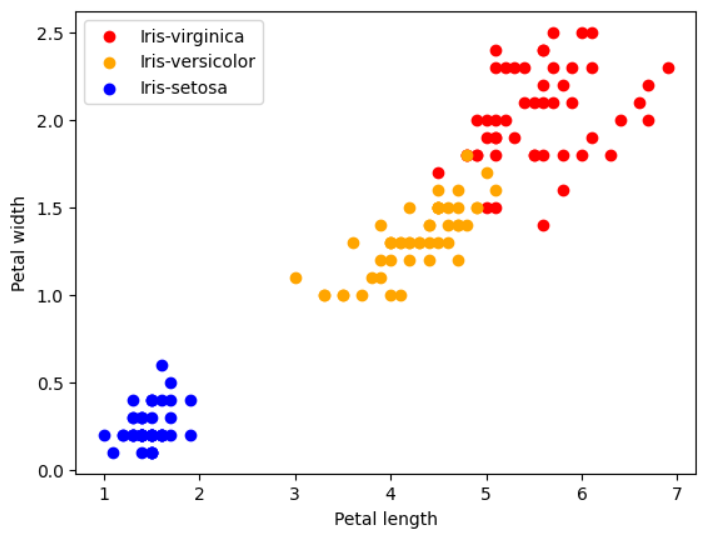
    x = data[data['species'] == species[i]]

    plt.scatter(x['petal-length'], x['petal-width'], c = colors[i], label=species[i])

    plt.xlabel("Petal length")

    plt.ylabel("Petal width")

    plt.legend()



for i in range(3):

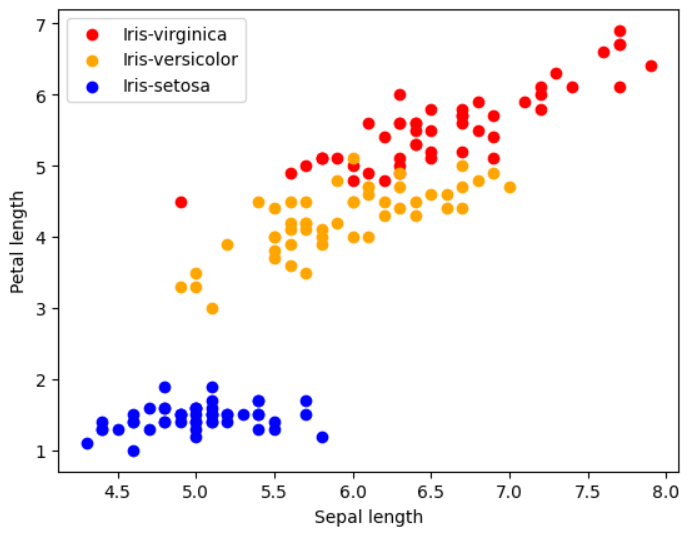
    x = data[data['species'] == species[i]]

    plt.scatter(x['sepal-length'], x['petal-length'], c = colors[i], label=species[i])

    plt.xlabel("Sepal length")

    plt.ylabel("Petal length")

    plt.legend()



for i in range(3):

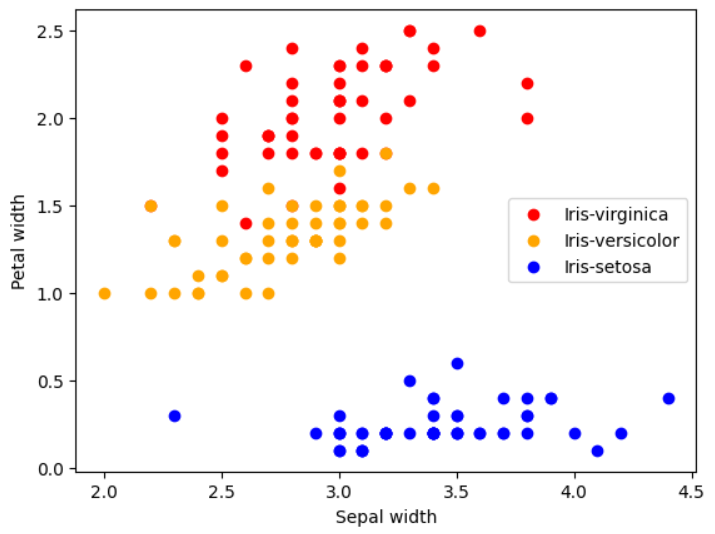
    x = data[data['species'] == species[i]]

    plt.scatter(x['sepal-width'], x['petal-width'], c = colors[i], label=species[i])

    plt.xlabel("Sepal width")

    plt.ylabel("Petal width")

    plt.legend()

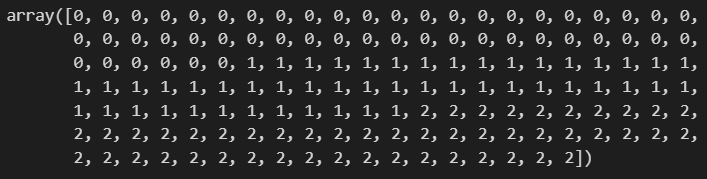


#Data pre-processing

from sklearn.preprocessing import LabelEncoder

label\_encoder=LabelEncoder()

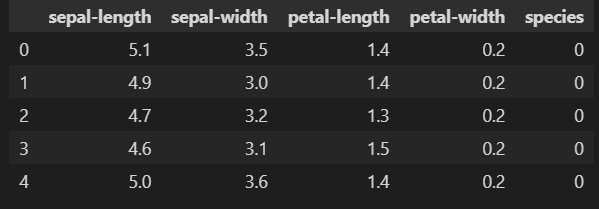
label\_encoder.fit\_transform(data['species'])



data['species']=label\_encoder.fit\_transform(data['species'])

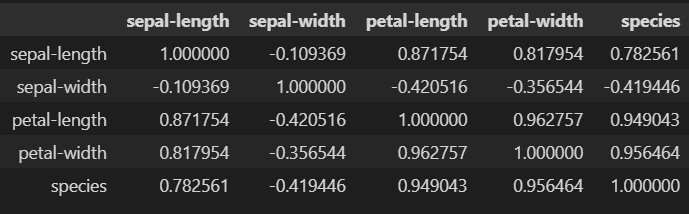
data.head()

#label encoding for string value-- string lai int value ma change garne



#Coorelation Matrix

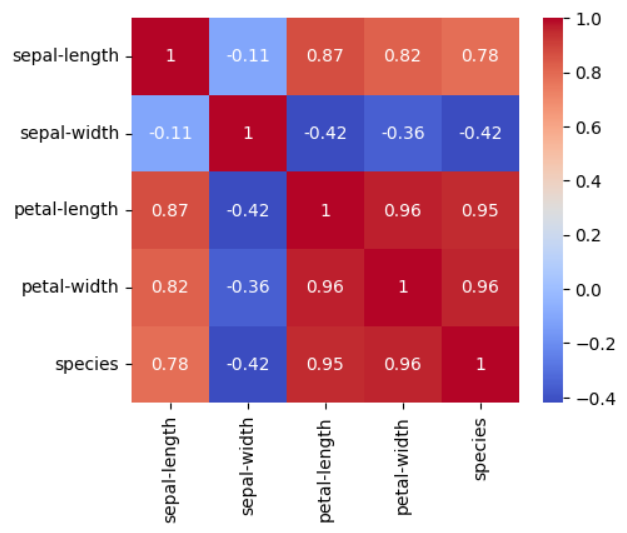
data.corr()



corr = data.corr()

fig, ax = plt.subplots(figsize=(5,4))

sns.heatmap(corr, annot=True, ax=ax, cmap='coolwarm')



#Model Training

from sklearn.model\_selection import train\_test\_split

#train - 70

#test - 30

X = data.drop(columns=['species'])

Y = data['species']

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X,Y, test\_size=0.30)

#Logistic regression

from sklearn.linear\_model import LogisticRegression

model = LogisticRegression()

#model training

model.fit(x\_train, y\_train)

#print metric to get performance

print("Accuracy:",model.score(x\_test, y\_test))



print("Accuracy:",model.score(x\_test, y\_test)\*100)

#knn - k-nearest neighbours

from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier()

model.fit(x\_train, y\_train)

print("Accuracy:",model.score(x\_test, y\_test)\*100)



#high level classifier

#decision tree

from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier()

model.fit(x\_train, y\_train)

print("Accuracy:",model.score(x\_test, y\_test)\*100)



# Conclusion

I embarked on a journey of data classification using Python in this report, focusing on the well-known Iris flower dataset. My primary goal was to create and test a classification model that could differentiate between three distinct Iris flower species: Iris setosa, Iris virginica, and Iris versicolor. I gained valuable insights and achieved the following key outcomes through meticulous implementation and analysis:

* Model Selection and Development: The model was trained on a large dataset of sepal and petal measurements, which allowed it to learn patterns and relationships in the data.
* Model Performance and Evaluation: I rigorously evaluated the model's performance using a variety of classification metrics.
* Implications for Practice: The successful classification of Iris flowers has practical applications in botany, horticulture, and ecological research. Accurate species identification can help researchers better understand and protect plant diversity.
* Visualization and Interpretation: To clarify the model's classification results and facilitate understanding, I used visual aids such as a confusion matrix. The graphical representation of the model's performance improves the understandability of the results.
* Future Work and Recommendations: To potentially improve classification accuracy, I recommend continuing to investigate alternative machine learning algorithms, hyperparameter tuning, and feature engineering.

Finally, this project serves as an example of data classification using Python, demonstrating not only the technical aspects of implementing a machine learning model, but also the critical evaluation and interpretation of results. The Iris flower dataset, with its simplicity and well-understood characteristics, is an excellent starting point for those interested in machine learning and data science. As I close this report, I want to emphasize the importance of this fundamental exercise and its potential for broader applications in data-driven decision-making and scientific research.