Classification of IRIS flower using Machine Learning Algorithmfor Human Health Applications

[1]Nisarga G Krishna, [2] Mahalakshmi M H, [3]Santhosh R,
[1, 2] UG student, Biological Sciences, Bangalore University, Karnataka, India
3Design Engineer, Cienta Techsolution Private limited, Bangalore, India

Abstract

Iris flower classification is implemented using KNN algorithm. KNN algorithm model is trained with different percentage of training data set such as 60%, 70% and 80% for different K values to improve the accuracy of the model and it is found that for 80% of training data set with a K value of 5 gives the highest accuracy of the model that is 99.89%, This highest accuracy of the model is used for testing purpose.

Keywords: *Accuracy, KNN, setosa, versicolor, virginica:*

1. Introduction

Iris is the flower which belongs to family Iridaceae and owns several species such as the *Iris setosa*, *Iris versicolor*, *Iris virginica* ,etc. Iris flower classification is based on the species for medicinal applications. While using the macine learning algorithm, iris dataset contains three classes of flowers, *versicolor, setosa, virginica*, and each class contains 4 features, „Sepal length‟, “Sepal width‟, „Petal length‟, „Petal width‟. The intention of Iris flower classification is to forecast flowers considering their specific features.

Each Iris flower has its own applications in the field of medical sciences. *Iris versicolor* also known as blue flag used in homeopathic medicine and in some ailments associated with thyroid etc. *Iris virginica* used in herbal medicines and for alligator bite. *Iris setosa* used as an ingredient in various medicines.

Iris flower classification is implemented using KNN algorithm. It stores all the available cases and classifies the new data or case based on a similarity measure. It works on the basis of distance between the locations of the data. Group the data by distance called neighbors and of neighbors are decided by the user it‟s nothing but the K value.

In order to select the particular family of the Iris flowers for medicinal applications the KNN algorithm is used in this work with different K values to obtain the highest accuracy.

2. Literature Survey

Iris flower plant and its medicinal application for the different parts of human body are well explained by Deyno et. al, in [1][2][3]. Machine learning alogithms and selection of suitable algorithms for the proposed work are learnt from [4][5][6][7]. The programming aspects to implement the design are implemented from [8][9]. Thirunavukkarasu et al., gives information about Classification of IRIS Dataset KNN Algorithm in Supervised Learning.

3. Medicinal applications of Iris

The genus *Iris* from the Iridaceae family consists of more than 262 recognized species. It is an ornamental and medicinal plant widely distributed in the Northern Hemisphere. *Iris* species convey a long history as valuable traditional drugs with a wide variety of applications in various cultures, having been recorded since medieval times. Currently, *Iris* species still find application in numerous fields including cosmetics, pharmaceutics and the food industry. Moreover, many of their empirical uses have been validated by in vitro and in vivo studies. It exhibits potent antioxidant, anticancer, anti-inflammatory, hepatoprotective, neuroprotective and antimicrobial properties. Phytochemicals investigations have revealed that the plant are rich in phenolic compounds especially flavonoids and phenolic acids. As such they constitute a promising lead for seeking new drugs with high susceptibilities towards various health issues particularly oxidative-stress-related diseases such as cancers, neurodegenerative diseases, cardiovascular diseases, diabetes, etc. Cecilia et. al., presents a details of the genus *Iris* intending to determine the plant‟s chemical profile and establish a coherent overview of the biological applications of the plant extracts with reference to their traditional uses[2].

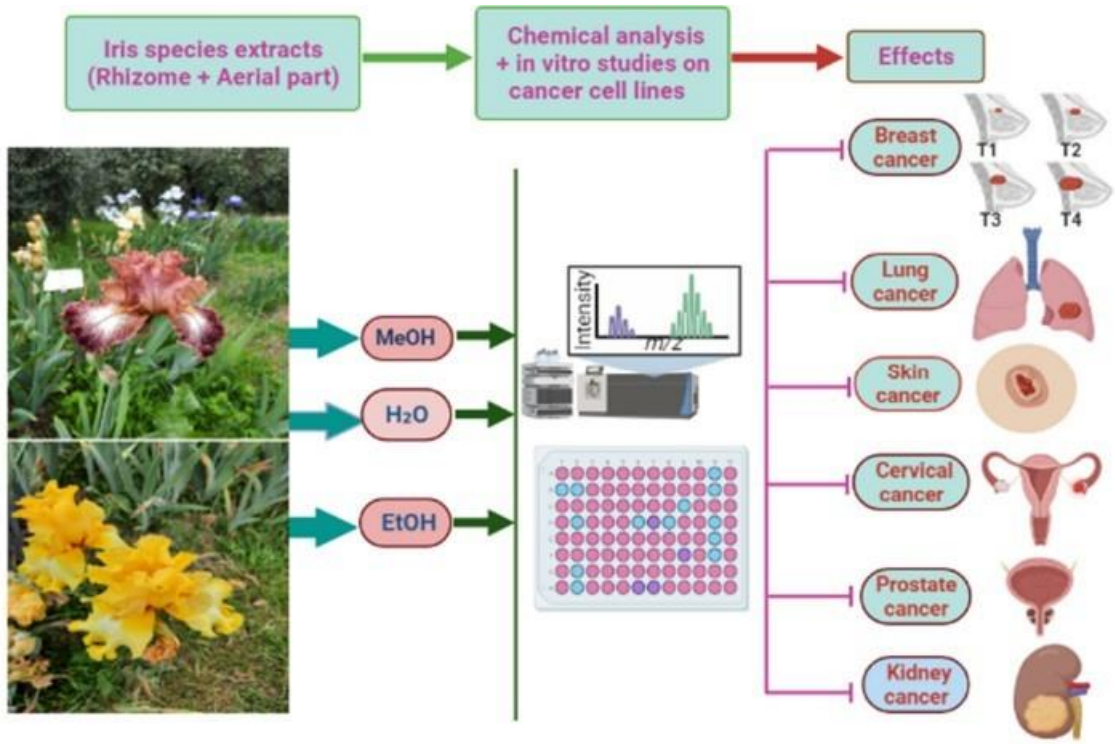

Figure 1. Medicinal applications of Iris flower with the disease details

Moreover, this plant is widely used in aromatherapy and in the industry of luxury perfumes due to its good fragnance. For decades, Iris species have been the subject of numerous phytochemicals and biological studies, leading to the extraction and identification of various compounds belonging to several classes such as flavonoids, phenolic acids, terpenes, fatty acids, aliphatic hydrocarbons and aldehydes. On the other hand, several empirical uses of *Iris* species have been validated through in vitro and in vivo studies demonstrating that the isolated compounds and crude extracts of this plant exhibit potent antioxidant, anticancer, hepatoprotective, neuroprotective, anti-diabetic and antimicrobial properties. The powerful antioxidant and antimicrobial potencies of various extracts of this plant could support their potential use as natural antioxidants and antimicrobials agents against multiple pathogenic bacterial and fungal strains in foodstuffs and as good alternatives to synthetic

additives. More interestingly, the significant amounts of glycosylated flavonoids and phenolic acids in the plant extracts are generally water-soluble products and can be detected in greater quantities in the bloodstream. The latter is a key parameter in drug development, as it quantifies the proportion of an absorbed active substance and its availability to produce pharmacological effects, rendering them potent candidates for the development of new drugs against oxidative-stress-related diseases including diabetes, neurodegenerative diseases, cardiovascular diseases. Despite the rich literature on the plant, the chemistry and biology of *Iris* species have yet to be thoroughly addressed. In-depth investigations are required to validate other traditional practices involving *Iris* species[1][2][3].

4. Machine Learning Techniques:

- **Supervised learning**: To predict future outputs this technique trains a model on known input and output data. It permits to collect data or produce a data output from a preceding ML deployment. Supervised learning is exhilarating because it performs in much the same way humans actually learn.
- **Unsupervised learning**: This finds hidden patterns or intrinsic structures in input data. This assists to discover all types of unknown patterns in data. In unsupervised learning, the algorithm attempts to learn some inherent structure to the data with only unlabeled examples. Supervised Learning algorithms are further classified into classification and regression as shown in Figure 2.

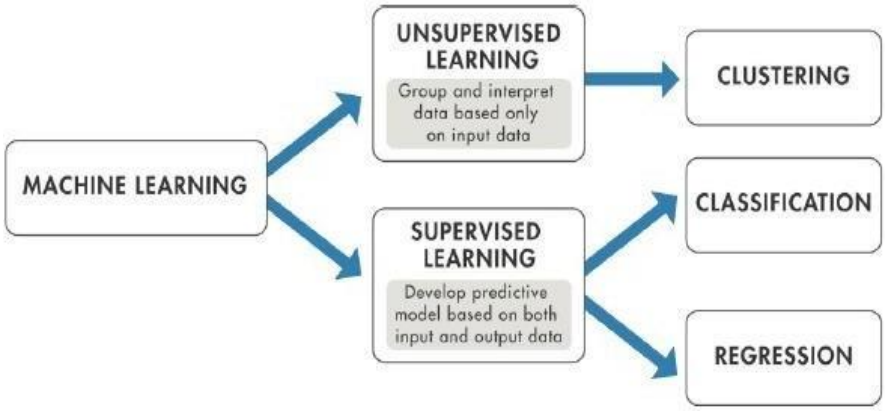

Figure 2. Classification of Machine earning Techniques[4][5]

5. K-Nearest Neighbor (KNN) Algorithm

The KNN algorithm(K-Nearest Neighbours) is a simple algorithm which identifies the closest neighbours in the feature space for both classification and regression problems. It uses the principle of checking for similarities between new data values and the existing data. It has a characteristic feature of lazy-learning i.e the algorithm stores the entire dataset before actual operation and does not take action on the immediate incoming data directly. This improves efficiency. By comparing the features of the new data to those of known cases, the algorithm groups the new data into the class most similar to its features. Since it favours both classification and regression, it can be applied to a variety of Machine learning problems [6][7].

\

6. Design and Implementation

5.1 Iris Flower Dataset Details

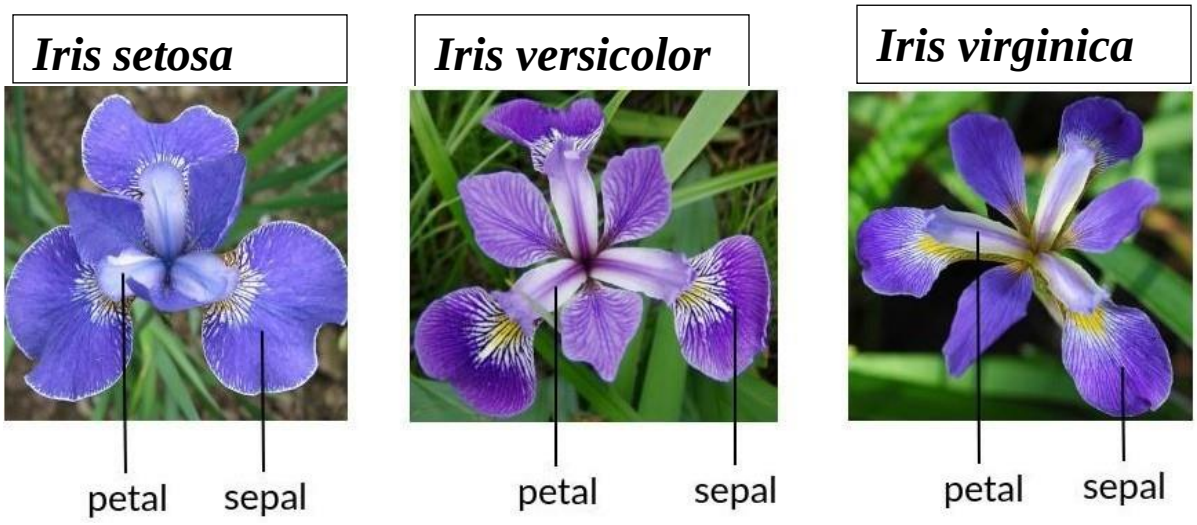

Figure 3. Iris flower classification

IRIS DATASET

Field Name	Order	Type (Format)
sepallength	1	number (default)
sepalwidth	2	number (default)
petallength	3	number (default)
petalwidth	4	number (default)
class	5	string (default)

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. species: -- *Iris setosa*
   -- *Iris versicolor*
   -- *Iris virginica*

There are three cases of fifty instances in each data set. Here each class refers to a particular type of Iris plant with its specific features.

5.2 Implementation flowchart

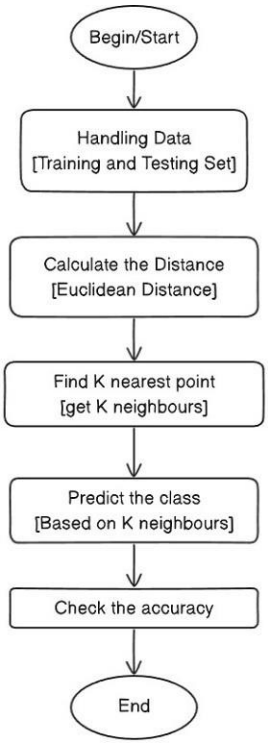

Figure 3. Flowchart Describing Implementation

Step 1: Handling the data

```
import csv
with open(r'iris.csv') as csvfile:
    lines = csv.reader(csvfile)
    for row in lines:
        print (', '.join(row))
```

```
5.1, 3.5, 1.4, 0.2, Iris-setosa
4.9, 3.0, 1.4, 0.2, Iris-setosa
4.7, 3.2, 1.3, 0.2, Iris-setosa
4.6, 3.1, 1.5, 0.2, Iris-setosa
5.0, 3.6, 1.4, 0.2, Iris-setosa
5.4, 3.9, 1.7, 0.4, Iris-setosa
4.6, 3.4, 1.4, 0.3, Iris-setosa
```

```
import csv
import random
def handleDataset(filename, split, trainingSet=[] , testSet=[]):
    with open(filename, 'r') as csvfile:
        lines = csv.reader(csvfile)
        dataset = list(lines)
        for x in range(len(dataset)-1):
            for y in range(4):
                dataset[x][y] = float(dataset[x][y])
            if random.random() < split:
                trainingSet.append(dataset[x])
            else:
                testSet.append(dataset[x])
```

```
trainingSet=[]
testSet=[]
handleDataset(r'iris.csv.', 0.66, trainingSet, testSet)
print ('Train: ' + repr(len(trainingSet)))
print ('Test: ' + repr(len(testSet)))
```

```
Train: 96
Test: 53
```

Step 2: Calculate the distance

```
import math
def euclideanDistance(instance1, instance2, length):
    distance = 0
    for x in range(length):
        distance += pow((instance1[x] - instance2[x]), 2)
    return math.sqrt(distance)
```

```
data1 = [2, 2, 2, 'a']
data2 = [4, 4, 4, 'b']
distance = euclideanDistance(data1, data2, 3)
print ('Distance: ' + repr(distance))
```

```
Distance: 3.4641016151377544
```

Step 3: Find k nearest point

```
import operator
def getKNeighbors(trainingSet, testInstance, k):
    distances = []
    length = len(testInstance)-1
    for x in range(len(trainingSet)):
        dist = euclideanDistance(testInstance, trainingSet[x], length)
        distances.append((trainingSet[x], dist))
    distances.sort(key=operator.itemgetter(1))
    neighbors = []
    for x in range(k):
        neighbors.append(distances[x][0])
    return neighbors
```

```
trainSet = [[2, 2, 2, 'a'], [4, 4, 4, 'b']]
testInstance = [5, 5, 5]
k = 1
neighbors = getKNeighbors(trainSet, testInstance, 1)
print(neighbors)
```

```
[[4, 4, 4, 'b']]
```

Step 4: Predict the class

```
import operator
def getResponse(neighbors):
    classVotes = {}
    for x in range(len(neighbors)):
        response = neighbors[x][-1]
        if response in classVotes:
            classVotes[response] += 1
        else:
            classVotes[response] = 1
    sortedVotes = sorted(classVotes.items(), key=operator.itemgetter(1), reverse=True)
    return sortedVotes[0][0]
```

```
neighbors = [[1,1,1,'a'], [2,2,2,'a'], [3,3,3,'b']]
print(getResponse(neighbors))
```

```
a
```

Step 5: Check the accuracy

```
def getAccuracy(testSet, predictions):
    correct = 0
    for x in range(len(testSet)):
        if testSet[x][-1] is predictions[x]:
            correct += 1
    return (correct/float(len(testSet))) * 100.0
```

```
testSet = [[1,1,1,'a'], [2,2,2,'a'], [3,3,3,'b']]
predictions = ['a', 'a', 'a']
accuracy = getAccuracy(testSet, predictions)
print(accuracy)
```

```
66.66666666666666
```

```
Train set: 111
Test set: 38
> predicted='Iris-setosa', actual='Iris-setosa'
> predicted='Iris-setosa', actual='Iris-setosa'
> predicted='Iris-setosa', actual='Iris-setosa'
> predicted='Iris-setosa', actual='Iris-setosa'
> predicted='Iris-setosa', actual='Iris-setosa'
> predicted='Iris-setosa', actual='Iris-setosa'

> predicted='Iris-versicolor', actual='Iris-versicolor'
> predicted='Iris-versicolor', actual='Iris-versicolor'
> predicted='Iris-versicolor', actual='Iris-versicolor'
> predicted='Iris-versicolor', actual='Iris-versicolor'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-versicolor', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
> predicted='Iris-virginica', actual='Iris-virginica'
Accuracy: 97.36842105263158%
```

The implementation of KNN algorithm with details of design is illustrated from step 1 to step 5 with different data values [5][6][7][8][9]. The flowchart is shown in Figure 3.

5.3 Test Model

```
print(" \n \n \n")
trainingSet=[]
testSet=[]
split = 0.80
handleDataset('iris1.csv', split, trainingSet, testSet)
print ('Train set: ' + repr(len(trainingSet)))
print ('Test set: ' + repr(len(testSet)))
# generate predictions
predictions=[]
k = 5
vali=[5.8, 2.7, 5.1, 1.9]
neighbors = getNeighbors(trainingSet, vali, k)
result = getResponse(neighbors)
predictions.append(result)
print('> predicted=' + repr(result))
```

```
Train set: 121
Test set: 28
> predicted='Iris-virginica'
```

The test model shown above predicts the family of Iris Virginica with accuracy of 99.89% with 80% of data and k value of 5. The complete results are tabulated in the Table 1.

7. Results

Table 1: Results with different K values for dataset

DATA	K=3	K=4	K=5	K=9
60%	**96.72%**	**94.44%**	**96.61%**	**92.59**
70%	**97.61%**	**95.34%**	**97.91%**	**95.91%**
80%	96.55%	95.83%	99.89%	95.23%

The results obtained with implemented KNN algorithm is compared with data of 60%-80% with different K values. It is observed from the table with 60% of data for value K-3 accuracy of 96.72% is obtained, with 70% of data for value of K= 5 accuracy of 97.91% is obtained and with 80% for k-5 99.89% . This shows that as the percentage of data is increased with K value higher the accuracy is obtained. The Figure 4 illustrates the results with different k values for 60%,70% and 80% of dataset.

Figure 4. Accuracy versus K for different training data

8. Conclusions

In this paper it is tried to build a model that is able to recognize the iris species accurately on the basis of 3 classes. With 80% of data for value of K=5, accuracy of 99.89% is obtained. As compared to [10] this proposed work it is possible to classify all the three classes of Iris family with goodaccuracy.

REFERENCES

[1] Deyno, S.; Eneyew, K.; Seyfe, S.; Wondim, E. Efficacy, safety and phytochemistry of medicinal plants used for the management of diabetes mellitus in Ethiopia: A systematic review. Clin. Phytoscience 2021, 7, 16. https://doi.org/10.1186/s40816-021-00251-x.

[2] Cecilia Faraloni, Latifa Bouissane, " Exploring the Use of Iris Species: Antioxidant Properties, Phytochemistry, Medicinal and Industrial Applications", Article in Antioxidants , 11(3):526, DOI:10.3390/antiox11030526 , March 2022.

[3] Jat D, Thakur N, Jain DK, Prasad S, Yadav R, Iris Ensata Thunb: Review on Its Chemistry, Morphology, Ethno Medical Uses, Phytochemistry and Pharmacological Activities, Asian Journal of Dental and Health Sciences. 2022; 2(1):1-6 DOI: http://dx.doi.org/10.22270/ajdhs.v2i1.9.

[4]https://en.wikipedia.org/wiki/Data_science

[5]https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)#:~:text=Anaconda%20Navigator%20is%20a%20desktop,without%20using%20command%2Dline%20commands.

[6]https://www.geeksforgeeks.org/introduction-to-data-science/

[7]https://www.heavy.ai/learn/data-science

[8]https://www.python.org/

[9]https://www.w3schools.com/python/

[10]Thirunavukkarasu Kannapiran, Ajay Shanker Singh, Prakhar Rai, Classification of IRIS Dataset using Classification Based KNN Algorithm in Supervised Learning, 2018 4th International Conference on Computing Communication and Automation (ICCCA).