MOHAMMED YOUNUS

Prediction of IRIS flower using Machine Learning Model.

Problem Statement – In this project we make prediction of the physical parameters of three species of Iris flower — Iris-Versicolor, Iris-Setosa and Iris-Virginica. The numeric parameters which the dataset use contains are Sepal width, Sepal length, Petal width and Petal length. In this Project we will be predicting the classes of the flowers based on these parameters. The data consists of continuous numeric values which describe the dimensions of the respective features. We will be training the model based on these features.

- Steps to be taken in the Project is sub-divided into the following sections. These are:

- Load the necessary libraries such as Numpy , Pandas , sklearn etc.
- Loading the dataset as csv file and showing first ten rows.
- Drop the unnecessary columns from the data.
- Calculate statistical values and round them up to 3 decimal places.
- Checking for null values and return their sum of numbers of true values in each column.
- Handle the null by mean of all values fill into them.
- Extracting all information about data.
- Checking shape of data and checking unique values in dependent variable.
- Visualization on different species of Iris flower using Python data visualization.
- Data preprocessing or (Data cleaning) performed by the one hot encoding in this process we change categorical data into numerical data and the technique is called feature Engineering.
- Splitting the cleaned data into dependent and independent variables.
- Splitting the data into train and test sets with train_test_split using sklearn library.
- Import different kind of Classification Models and Train that model with the help of .fit().
- Predicting the trained models and then checking their accuracy score and confusion metrics of the model using confusion metrics & accuracy score.
- Then recall the train_test_split and split the data into training and testing set with different models.
- Then predicting the trained models and checking the accuracy of model and check the accuracy difference.
- And finally predict whether the classification of different species of Iris is generated or not.

- Step-1 - Loading Necessary Libraries used in machine learning.

Import Necessary Libraries.

```
[65] import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as snr
     from sklearn.preprocessing import OneHotEncoder
```

- Step-2 - Loading the dataset as csv file and showing first ten rows.

Load data and show first 10 rows of data.

```
data=pd.read_csv("/content/IRIS.csv")
data.head(10)
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

- Step-3 - Calculate statistical values and round them up to 3 decimal places.

Calculates statistical values and rounds them to 3 decimal places.

```
data.describe().round(3)
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000	150.000	150.000	150.000
mean	5.843	3.054	3.759	1.199
std	0.828	0.434	1.764	0.763
min	4.300	2.000	1.000	0.100
25%	5.100	2.800	1.600	0.300
50%	5.800	3.000	4.350	1.300
75%	6.400	3.300	5.100	1.800
max	7.900	4.400	6.900	2.500

+ Code + Text

- Step-4 – Checking for null values and return their sum of numbers of true values in each column.

Mark null values as True and returns sum of number of True values in each column.

```
[4] data.isnull().sum()
```

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

- Step-5 – Extracting all information about data.

Extracting all information about data

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

- Step-6 - Checking shape of data and checking unique values in dependent variable.

Shape of data

```
[6] data.shape
```

```
(150, 5)
```

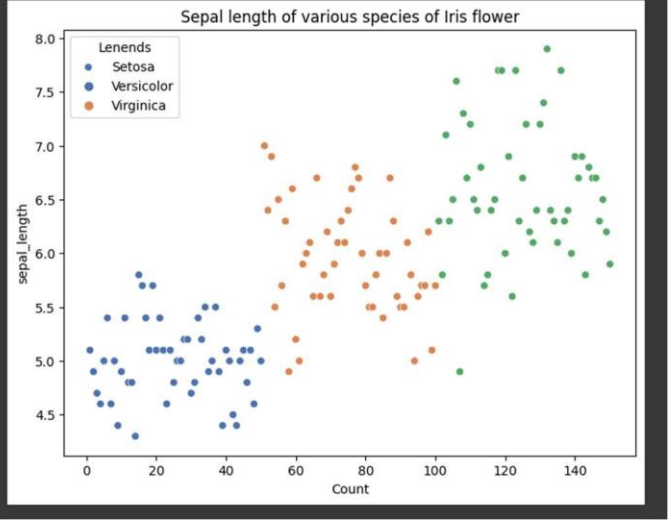
Checking Unique values of species column.

```
[7] data['species'].unique()
```

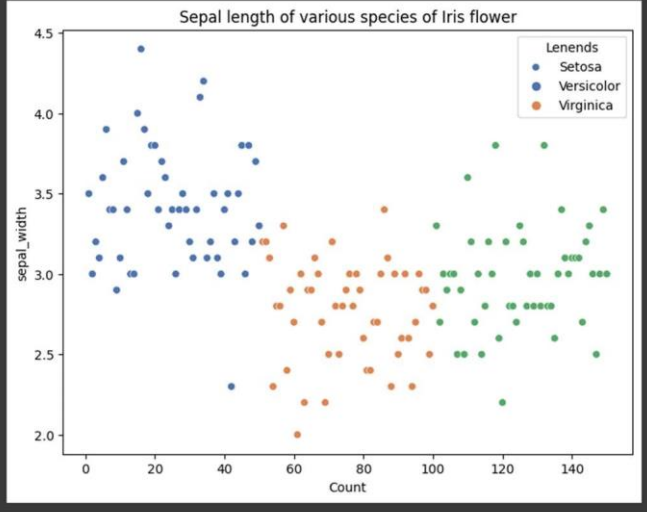
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

Step-7 - Visualization on different species of Iris flower using Python Data Visualization.

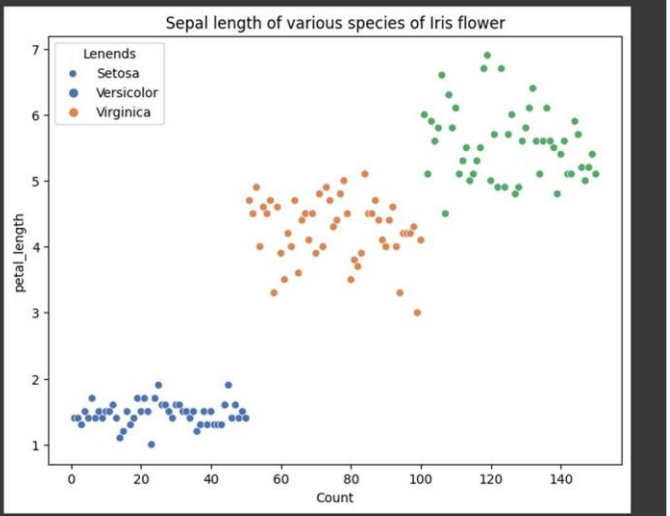
```
plt.figure(figsize=(8,6))
snr.scatterplot(x='sno', y='sepal_length', hue='species', palette='deep', data=data)
plt.xlabel('Count')
plt.title('Sepal length of various species of Iris flower')
plt.legend(title="Lenends", labels=['Setosa','Versicolor','Virginica'])
plt.show()
```



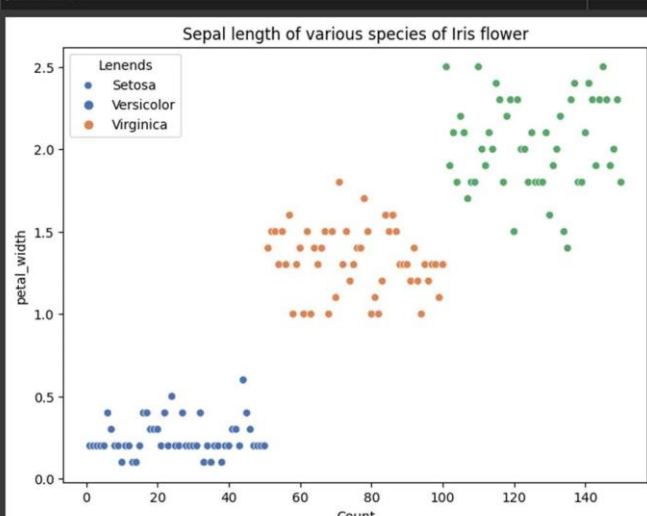

```
plt.figure(figsize=(8,6))
snr.scatterplot(x='sno', y='sepal_width', hue='species', palette='deep', data=data)
plt.xlabel('Count')
plt.title('Sepal length of various species of Iris flower')
plt.legend(title="Lenends", labels=['Setosa','Versicolor','Virginica'])
plt.show()
```




```
plt.figure(figsize=(8,6))
snr.scatterplot(x='sno', y='petal_length', hue='species', palette='deep', data=data)
plt.xlabel('Count')
plt.title('Sepal length of various species of Iris flower')
plt.legend(title="Lenends", labels=['Setosa','Versicolor','Virginica'])
plt.show()
```




```
plt.figure(figsize=(8,6))
snr.scatterplot(x='sno', y='petal_width', hue='species', palette='deep', data=data)
plt.xlabel('Count')
plt.title('Sepal length of various species of Iris flower')
plt.legend(title="Lenends", labels=['Setosa','Versicolor','Virginica'])
plt.show()
```



- Step-8 – Data preprocessing or (Data cleaning) performed by the one hot encoding in this process we change categorical data into numerical data and the technique is called feature Engineering.

Change the catagorical data into numerical data by using OneHotEncoding.

```
data['species']=data['species'].apply({'Iris-setosa':0,'Iris-versicolor':1,'Iris-virginica':2}.get)
data.head()
```

	sno	sepal_length	sepal_width	petal_length	petal_width	species
0	1	5.1	3.5	1.4	0.2	0
1	2	4.9	3.0	1.4	0.2	0
2	3	4.7	3.2	1.3	0.2	0
3	4	4.6	3.1	1.5	0.2	0
4	5	5.0	3.6	1.4	0.2	0

- Step-9 – Splitting the cleaned data into dependent and independent variables.

Deviding the data into Dependent and Independent variables.

```
x=data.drop(['species'],axis=1)
y=data['species']
x.head()
```

	sno	sepal_length	sepal_width	petal_length	petal_width
0	1	5.1	3.5	1.4	0.2
1	2	4.9	3.0	1.4	0.2
2	3	4.7	3.2	1.3	0.2
3	4	4.6	3.1	1.5	0.2
4	5	5.0	3.6	1.4	0.2

- Step-10 – Splitting the data into train and test sets with train_test_split using sklearn library.

Deviding the cleaned data into training and testing sets.

```
[12] from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.8)
```

- Step-11 – Import first machine learning model ‘Logistic regression’.

Creating first machine learning model 'Logistic Regression'.

```
[13] from sklearn.linear_model import LogisticRegression
     log=LogisticRegression()
```

Step-12 – Train the model using .fit() function.

Train the model.

```
[14] log.fit(x_train,y_train)

/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
▾ LogisticRegression
LogisticRegression()
```

Step-13 – Make predictions on model using .predict() function.

Make predictions on model

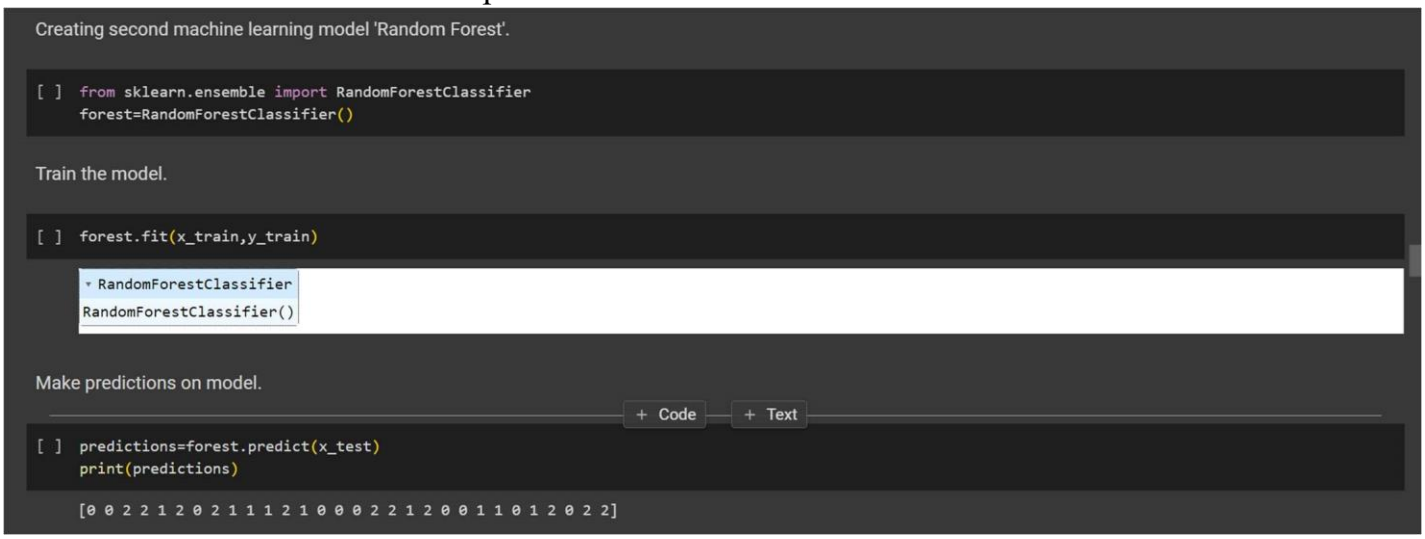
```
predictions=log.predict(x_test)
print(predictions)

[0 0 2 2 1 2 0 2 1 1 1 2 1 0 0 0 2 2 1 2 0 0 1 1 0 1 2 0 2 2]
```

Step-14 – Check the accuracy score and print a confusion metrics with confusion metrics & accuracy score.

Check confusion metrics and accuracy score.

```
[732] from sklearn.metrics import confusion_matrix, accuracy_score
      cm=confusion_matrix(y_test,predictions)
      ac=accuracy_score(y_test,predictions)

[733] print(cm)

      [[ 7  0  0]
       [ 0  8  0]
       [ 0  1 14]]

      print(ac)

      0.9666666666666667
```

Step-15 – Import the Second Machine Learning Model ‘Random Forest’ and train model and then make prediction.



Step-16 – Print a confusion metrics and check accuracy score for Random forest Model.

Shahjan Khan

Step-17 – Import the Third Machine Learning Model Support Vector Machine and train model and then make prediction.

Step-18 – Print a confusion metrics and check accuracy score for Support Vector Machine Model.

Conclusion – This project demonstrats the end-to-end process of building a machine learning model for Iris flower classification. It includes data preprocessing, exploratory data analysis, and model training. the first model is used in this project is Logistic Regression, second used is Random Forest, and third is Support vector Machine and model was trained with an accuracy of 97% approximate.

Thank you