Jayathilaka H.A.D.T.T.

E/16/156

CO542

Neural Networks and Fuzzy Systems

2021

Lab 04 - MLP

Task 3

1. Load the dataset as a Panda's dataframe and drop any unnecessary columns.

In [5]:
```
import matplotlib.pyplot as plt
from sklearn.neural_network import MLPClassifier
import pandas as pd
from sklearn.metrics import plot_confusion_matrix
import seaborn as sns
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report

#-----------------------------Question 1----------------------------------------

iris_data = pd.read_csv("iris.csv")
iris_data.head()
```

Out[5]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

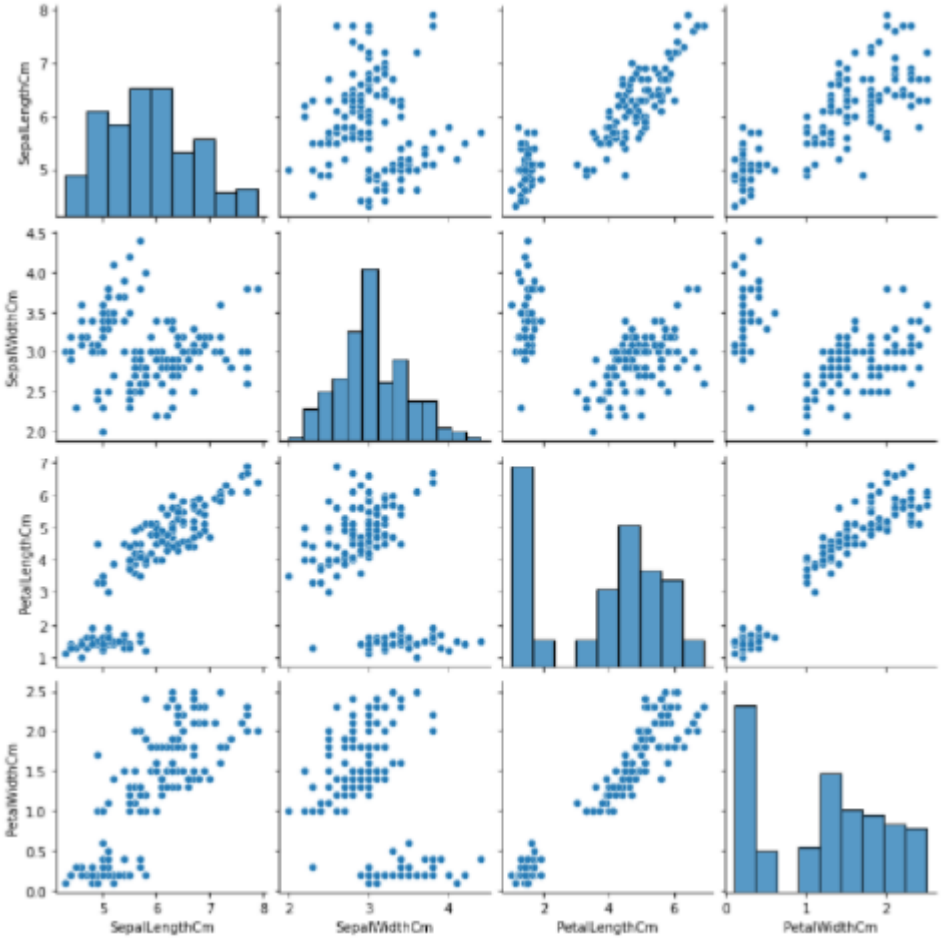
In [6]:
```
iris_data = iris_data.drop(['Id'], 1)
iris_data.head()
```

```
C:\Users\MYPC~1\AppData\Local\Temp/ipykernel_14528/1897028150.py:1: FutureWarning: In a future version of pandas all arguments
of DataFrame.drop except for the argument 'labels' will be keyword-only
  iris_data = iris_data.drop(['Id'], 1)
```

Out[6]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	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

2. Visualize the relationships between dataset features using seaborn.pairplot



3. Separate the dataset into features and labels.

In [9]:
```
feature_names = ['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']
features = iris_data[feature_names]
labels = iris_data['Species']
print(features.shape)
print(labels.shape)
```

```
(150, 4)
(150,)
```

4. **Convert categorical data in your dependent variable into numerical values using Sklearn Label Encoder.**

```
In [10]: label_encoder = preprocessing.LabelEncoder()
         label_encoder.fit(labels)
         labels = label_encoder.transform(labels)
```

5. **Split the data set as train and test data accordingly (E.g.: 80% training data, 20% test data).**

```
In [11]: x_train, x_test, y_train, y_test = train_test_split(features, labels, test_size=0.2,random_state=0)
         print(x_train.shape)
         print(y_train.shape)
```

```
(120, 4)
(120,)
```

6. **Scale the independent train and test datasets using Sklearn Standard Scaler.**

```
In [12]: s_f = StandardScaler()
         x_train = s_f.fit_transform(x_train)
         x_test = s_f.transform(x_test)
```

7. **Model the MLP using MLPClassifier.**

```
In [20]: model = MLPClassifier(hidden_layer_sizes=(32), activation="relu", random_state=1, max_iter=50)
```

8. **Make predictions using previously reserved test data set.**

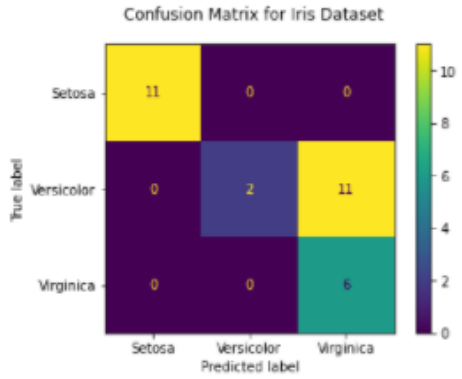
```
In [21]: result = model.fit(x_train, y_train)
         l_pred = result.predict(x_test)
         print(l_pred)
         print(result.score(x_test, y_test))
```

```
[2 2 0 2 0 2 0 2 2 2 2 2 2 2 0 2 2 0 0 2 1 0 0 2 0 0 2 1 0]
0.6333333333333333
```

9. **Observe the accuracy of the model by plotting the confusion metrix..**

```
In [24]: target_names = ["Setosa", "Versicolor", "Virginica"]
         fig = plot_confusion_matrix(result, x_test, y_test, display_labels=target_names)
         fig.figure_.suptitle("Confusion Matrix for Iris Dataset")
         plt.show()
```

```
C:\Users\My PC\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning: Functio
n plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is deprecated in 1.0 and will be removed in 1.2. Use on
e of the class methods: ConfusionMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
```



10. Generate the classification report using Sklearn Classification Report and comment on each of the value you obtained in that report.

```
In [25]: print(classification_report(y_test, l_pred, target_names=["Setosa", "Versicolor", "Virginica"]))
```

	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	1.00	0.15	0.27	13
Virginica	0.35	1.00	0.52	6
accuracy			0.63	30
macro avg	0.78	0.72	0.60	30
weighted avg	0.87	0.63	0.59	30

Macro average = (precision of setosa + Precision of versicolor + Precision of virginica) / 3

Weigted average = (all the true positives/ total number of objects in all classes)

Precision

- Precision = True positive / (True positive + False positive)
- Setosa has a precision of 1.00, indicating that all objects predicted as Iris-setosa are indeed setosa objects. When looking at the confusion matrix above, there are 11 objects that are expected to be setosa and are all setosa objects. As a result, 11/11 = 1.00.
- Versicolor's precision is 1.00, implying that all objects predicted as versicolor are truly versicolor objects. When looking at the aforementioned confusion matrix, there are 12 objects that are expected to be versicolor, and they are all versicolor objects. As a result, 12/12 = 1.00.
- Virginica has a precision of 0.86, which suggests that some objects predicted as virginica are not virginica. In the above confusion matrix, there are seven objects that are anticipated to be virginica, yet only six of them are. 6/7 = 0.86 as a result.

Recall

- Recall = True positive / (True positive + False negative)
- Setosa has a recall of 1.00, indicating that the model has never predicted a setosa object incorrectly.
- Because the model predicts one versicolor as a virginica object, the recall of versicolor is 0.92.
- Because no virginica object was predicted incorrectly by the model, the recall of virginica is 1.00.

F1-score

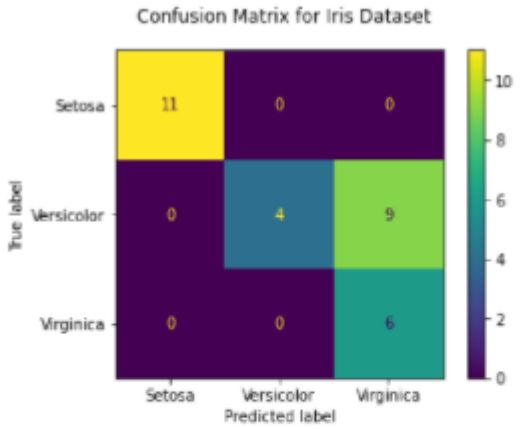
- F1-score = 2*Precision . Recall / (Precision + Recall)
- F1-score of class setosa is calculated using the above equation
- F1-score of class versicolor is calculated using the above equation
- F1-score of class virginica is calculated using the above equation

Support

- The number of actual occurrences of the class in the provided dataset is known as support.
- There are 11 occurrences of setosa. As a result, the number of support is 11.
- There are 13 occurrences of versicolor in the real world. As a result, the support is 13.
- virginica has actual 6 occurrences. As a result, the support is 6.

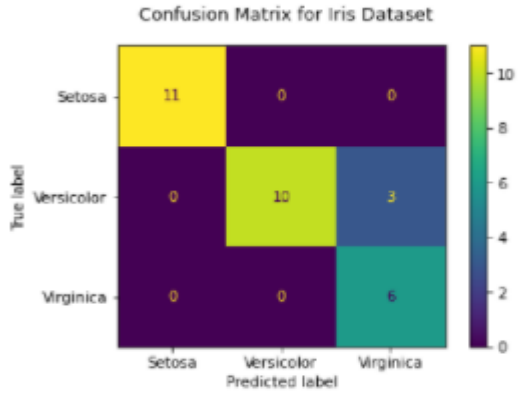
11. Repeat the task with 100,300,500 iterations and comment on your results.

When Iterations = 100



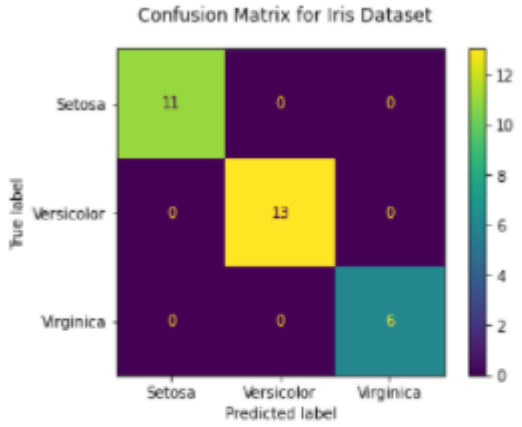
	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	1.00	0.31	0.47	13
Virginica	0.40	1.00	0.57	6
accuracy			0.70	30
macro avg	0.80	0.77	0.68	30
weighted avg	0.88	0.70	0.68	30

When Iterations = 300



	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	1.00	0.77	0.87	13
Virginica	0.67	1.00	0.80	6
accuracy			0.90	30
macro avg	0.89	0.92	0.89	30
weighted avg	0.93	0.90	0.90	30

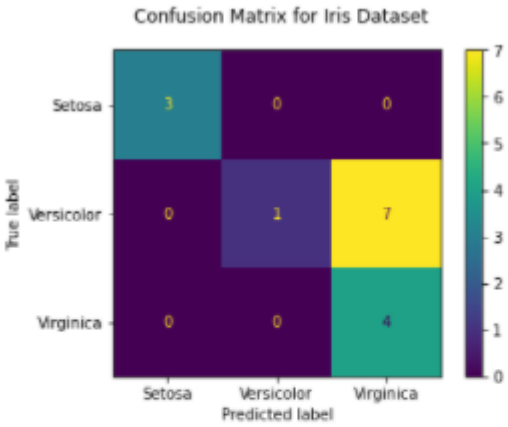
When Iterations = 500



	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	1.00	1.00	1.00	13
Virginica	1.00	1.00	1.00	6
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

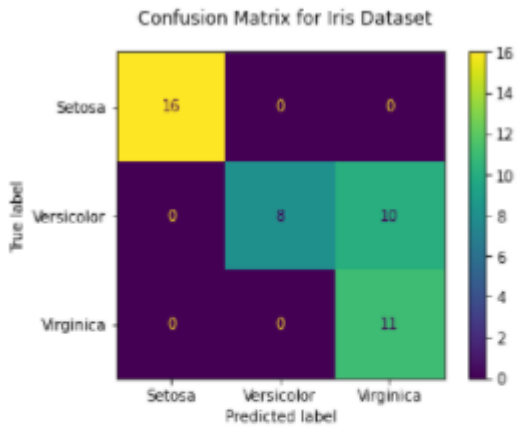
12. Repeat the task with varying test and train data set sizes and comment on your observations.

When test size=0.1



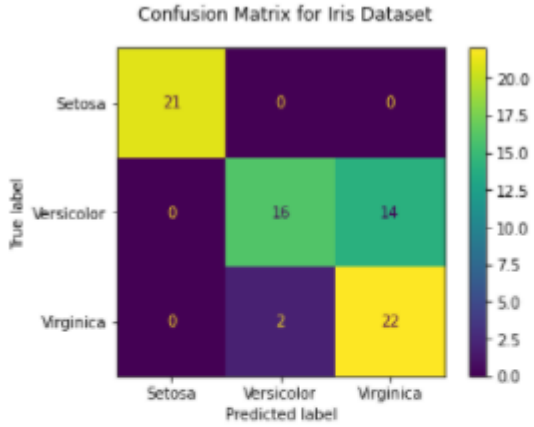
	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	3
Versicolor	1.00	0.12	0.22	8
Virginica	0.36	1.00	0.53	4
accuracy			0.53	15
macro avg	0.79	0.71	0.59	15
weighted avg	0.83	0.53	0.46	15

When test size=0.3



	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	16
Versicolor	1.00	0.44	0.62	18
Virginica	0.52	1.00	0.69	11
accuracy			0.78	45
macro avg	0.84	0.81	0.77	45
weighted avg	0.88	0.78	0.77	45

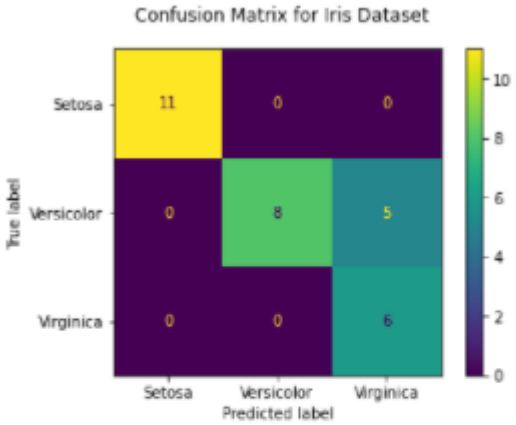
When test size=0.5



	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	21
Versicolor	0.89	0.53	0.67	30
Virginica	0.61	0.92	0.73	24
accuracy			0.79	75
macro avg	0.83	0.82	0.80	75
weighted avg	0.83	0.79	0.78	75

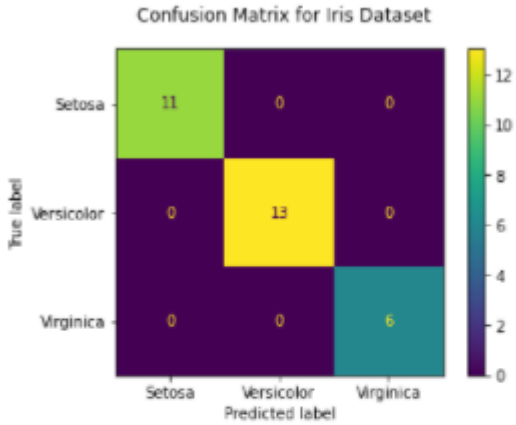
13. Repeat the task with different learning rates: 0.002, 0.5, 1.00 and comment on the results obtained.

When learning rate=0.002



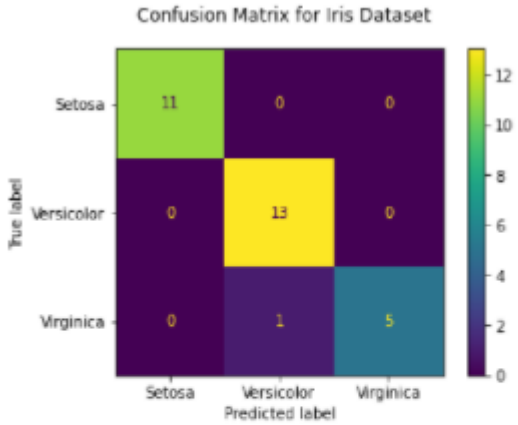
	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	1.00	0.62	0.76	13
Virginica	0.55	1.00	0.71	6
accuracy			0.83	30
macro avg	0.85	0.87	0.82	30
weighted avg	0.91	0.83	0.84	30

When learning rate=0.5



	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	1.00	1.00	1.00	13
Virginica	1.00	1.00	1.00	6
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

When learning rate=1



	precision	recall	f1-score	support
Setosa	1.00	1.00	1.00	11
Versicolor	0.93	1.00	0.96	13
Virginica	1.00	0.83	0.91	6
accuracy			0.97	30
macro avg	0.98	0.94	0.96	30
weighted avg	0.97	0.97	0.97	30