

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import preprocessing
```

Iris Classification Dataset

Input Features:

sepal_length, sepal_width, petal_length, petal_width

Target:

Iris plant class

Objective: Predict iris plant class for a given sepal_length, sepal_width, petal_length, petal_width

Data source: <https://archive.ics.uci.edu/ml/datasets/iris>

```
In [2]: columns = ['encoded_class', 'sepal_length', 'sepal_width', 'petal_length', 'petal_width']
```

```
In [3]: # Encode Class Labels to integers
le = preprocessing.LabelEncoder()
le.fit(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'])
```

```
Out[3]: ▼ LabelEncoder
LabelEncoder()
```

```
In [4]: le.classes_
```

```
Out[4]: array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype='<U15')
```

```
In [5]: df = pd.read_csv('iris_all.csv')
```

```
In [6]: df['class'].value_counts()
```

```
Out[6]: class
      Iris-setosa      50
      Iris-versicolor  50
      Iris-virginica   50
      Name: count, dtype: int64
```

```
In [7]: df.head()
```

```
Out[7]:
```

	sepal_length	sepal_width	petal_length	petal_width	class
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

```
In [8]: df.tail()
```

```
Out[8]:
```

	sepal_length	sepal_width	petal_length	petal_width	class
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

```
In [9]: le.transform(df['class'])[-5:]
```

```
Out[9]: array([2, 2, 2, 2, 2])
```

```
In [10]: # Convert Classes to numeric value
df['encoded_class'] = le.transform(df['class'])
```

```
In [11]: df.head()
```

Out[11]:

	sepal_length	sepal_width	petal_length	petal_width	class	encoded_class
0	5.1	3.5	1.4	0.2	Iris-setosa	0
1	4.9	3.0	1.4	0.2	Iris-setosa	0
2	4.7	3.2	1.3	0.2	Iris-setosa	0
3	4.6	3.1	1.5	0.2	Iris-setosa	0
4	5.0	3.6	1.4	0.2	Iris-setosa	0

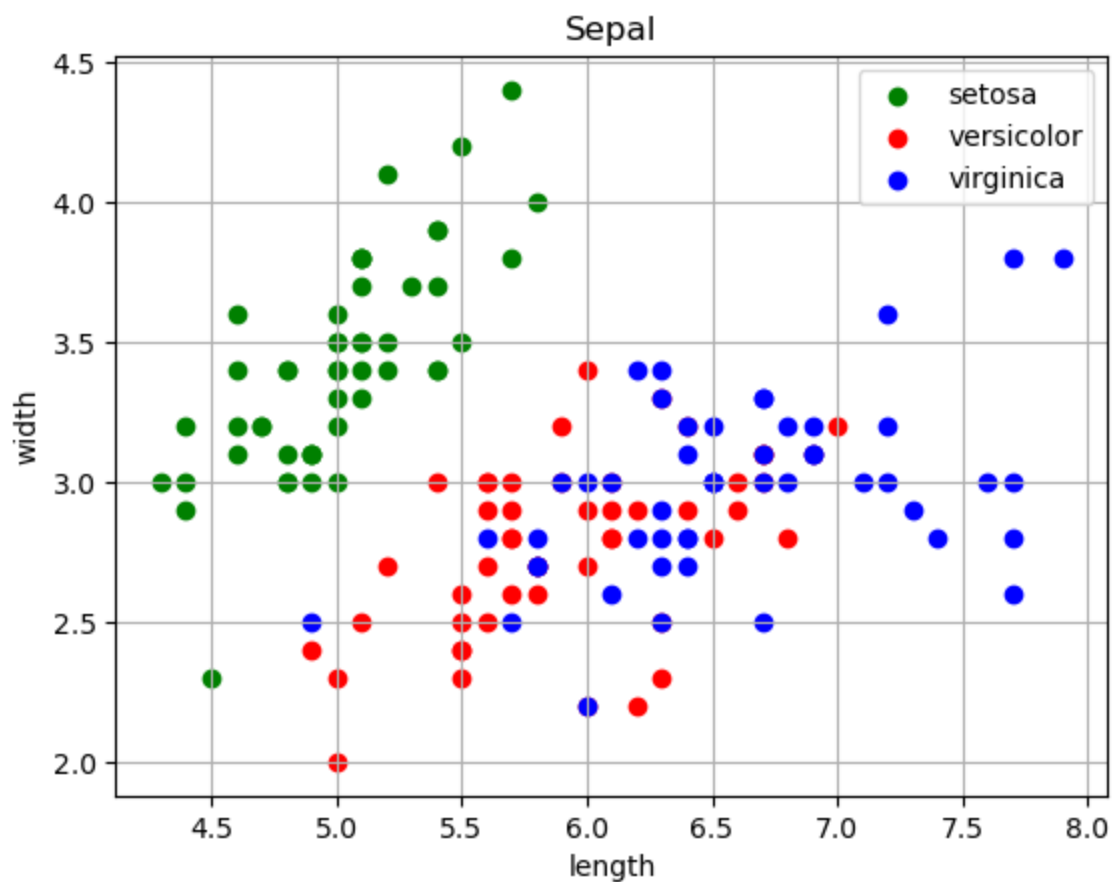
In [12]: `df.tail()`

Out[12]:

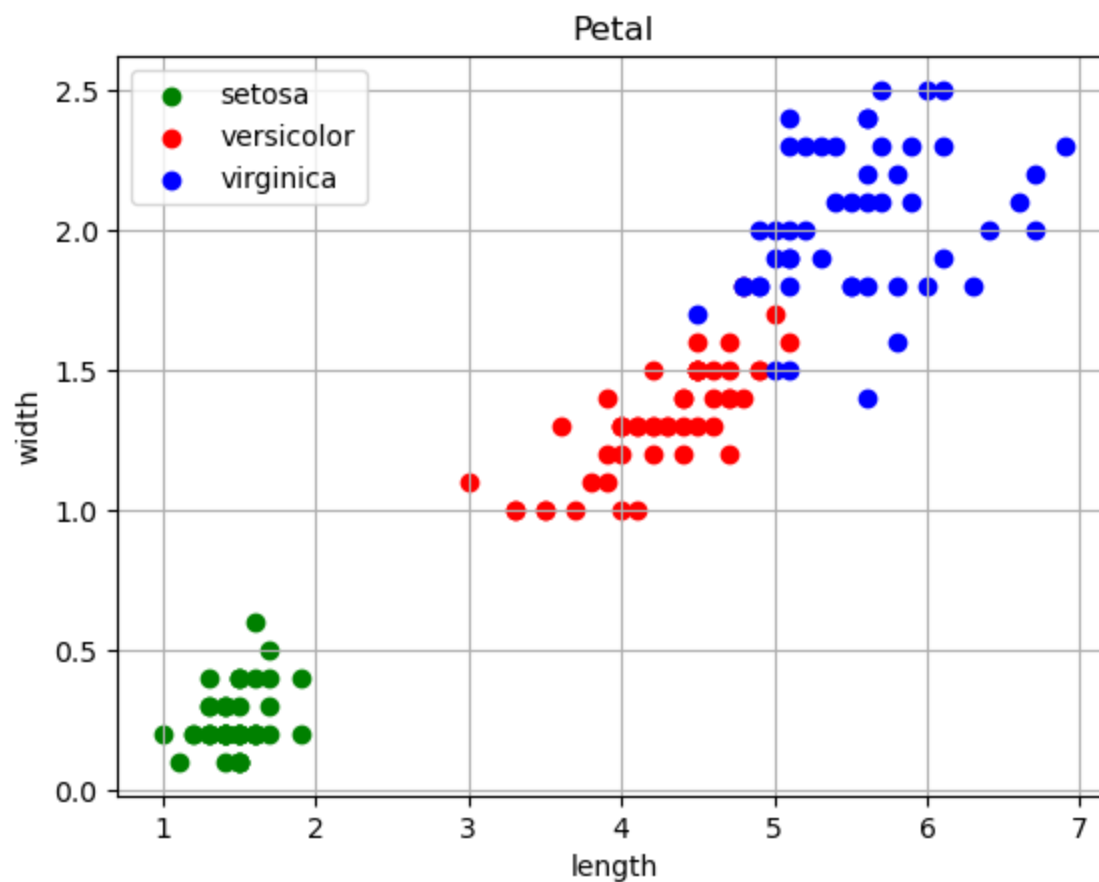
	sepal_length	sepal_width	petal_length	petal_width	class	encoded_class
145	6.7	3.0	5.2	2.3	Iris-virginica	2
146	6.3	2.5	5.0	1.9	Iris-virginica	2
147	6.5	3.0	5.2	2.0	Iris-virginica	2
148	6.2	3.4	5.4	2.3	Iris-virginica	2
149	5.9	3.0	5.1	1.8	Iris-virginica	2

In [13]: `# Visualize`
`setosa = df['class'] == 'Iris-setosa'`
`versicolor = df['class'] == 'Iris-versicolor'`
`virginica = df['class'] == 'Iris-virginica'`

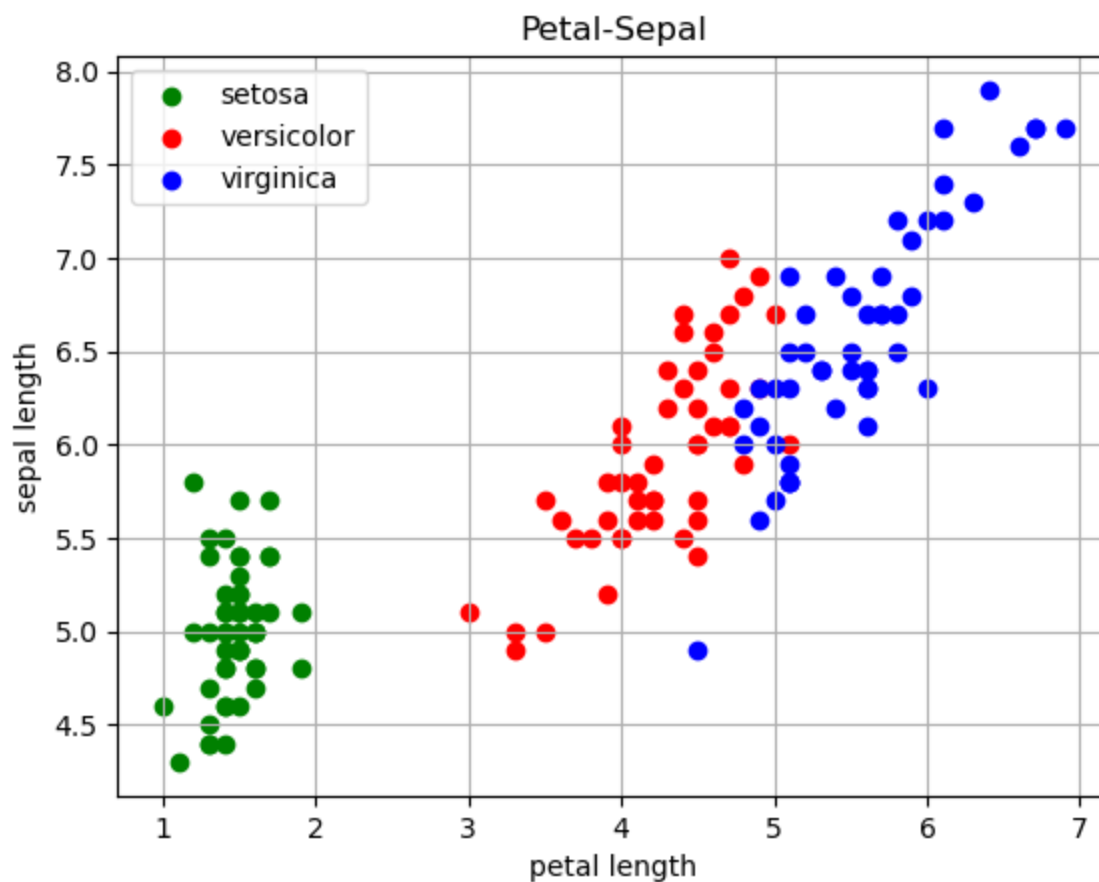
In [14]: `plt.scatter(df[setosa].sepal_length,y=df[setosa].sepal_width, label='setosa',color='g')`
`plt.scatter(df[versicolor].sepal_length,y=df[versicolor].sepal_width, label='versicolor',color='r')`
`plt.scatter(df[virginica].sepal_length,y=df[virginica].sepal_width, label='virginica',color='b')`
`plt.xlabel('length')`
`plt.ylabel('width')`
`plt.title('Sepal')`
`plt.grid(True)`
`plt.legend()`
`plt.show()`



```
In [15]: plt.scatter(df[setosa].petal_length,y=df[setosa].petal_width, label='setosa',color='g')
plt.scatter(df[versicolor].petal_length,y=df[versicolor].petal_width, label='versicolor',color='r')
plt.scatter(df[virginica].petal_length,y=df[virginica].petal_width, label='virginica',color='b')
plt.xlabel('length')
plt.ylabel('width')
plt.title('Petal')
plt.grid(True)
plt.legend()
plt.show()
```



```
In [16]: plt.scatter(df[setosa].petal_length,y=df[setosa].sepal_length, label='setosa',color='g')
plt.scatter(df[versicolor].petal_length,y=df[versicolor].sepal_length, label='versicolor',color='r')
plt.scatter(df[virginica].petal_length,y=df[virginica].sepal_length, label='virginica',color='b')
plt.xlabel('petal length')
plt.ylabel('sepal length')
plt.title('Petal-Sepal')
plt.grid(True)
plt.legend()
plt.show()
```



Training and Validation Set

Target Variable as first column followed by input features:

class,sepal_length,sepal_width,petal_length,petal_width

Training, Validation files do not have a column header

```
In [17]: # Training = 70% of the data  
# Validation = 30% of the data  
# Randomize the dataset
```

```
np.random.seed(5)
l = list(df.index)
np.random.shuffle(l)
df = df.iloc[l]
```

```
In [18]: rows = df.shape[0]
train = int(.7 * rows)
test = rows - train
```

```
In [19]: rows, train, test
```

```
Out[19]: (150, 105, 45)
```

```
In [20]: # Write Training Set
df[:train].to_csv('iris_train.csv'
                  , index=False, header=False
                  , columns=columns)
```

```
In [21]: # Write Validation Set
df[train:].to_csv('iris_validation.csv'
                  , index=False, header=False
                  , columns=columns)
```

```
In [22]: # Write Column List
with open('iris_train_column_list.txt', 'w') as f:
    f.write(','.join(columns))
```

```
In [ ]:
```