# 25. Data Science – Machine Learning – Support Vector Machine

## Contents

# 25. Data Science – Machine Learning – Support Vector Machine

## 1. Support Vector Machine

- ✓ Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms.
- ✓ It is used for Classification as well as Regression problems.
- ✓ Mostly using for Classification problems in Machine Learning.

## 2. Decision boundary

- ✓ The goal of the SVM algorithm is to create the best line or decision boundary that can separate n-dimensional space into classes.
- ✓ This best decision boundary is called a hyperplane.

## 3. Why SVM name is SVM

- ✓ SVM chooses the extreme points/vectors that help in creating the hyperplane.
- ✓ These extreme cases are called as support vectors; hence this algorithm is called as Support Vector Machine.

## 4. Usage

- ✓ SVM algorithm can be used for,
  - Face detection
  - Image classification
  - Text categorization

## 5. Types of SVM

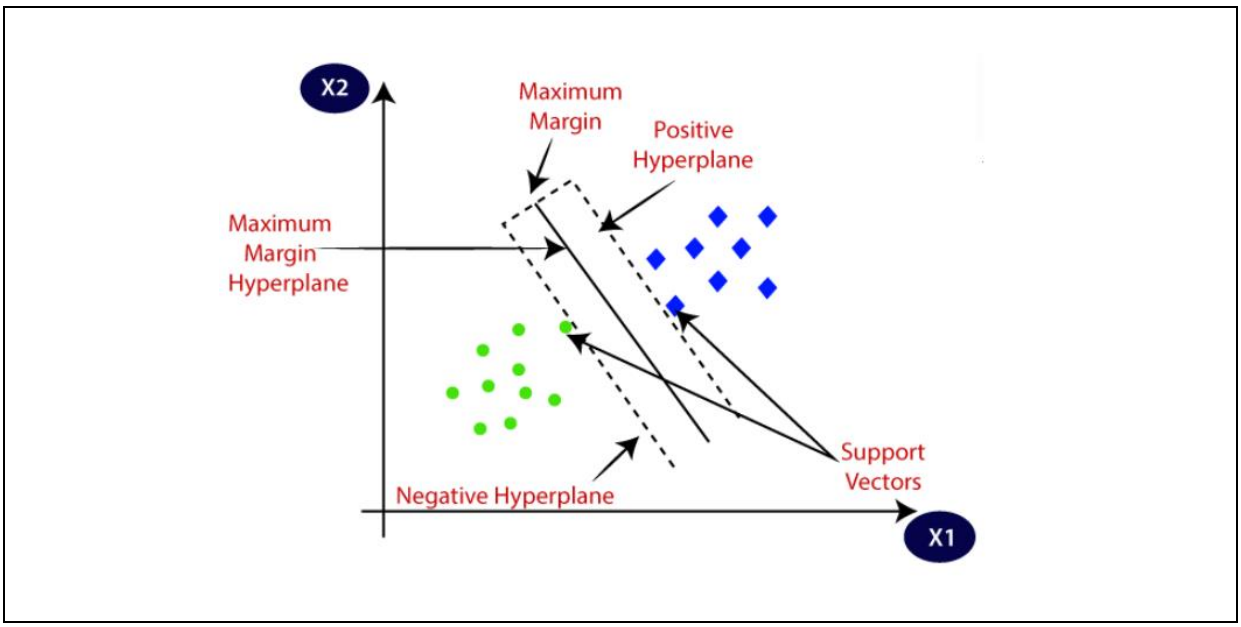
- ✓ Linear SVM
- ✓ Non-linear SVM

## 6. Linear SVM

- ✓ If the dataset can be classified into two classes by using a single straight line, then that is called as linearly separable data, and classifier is used called as Linear SVM classifier.

## 7. Non-linear SVM

- ✓ If the data set cannot be classified by using a straight line, then that is called as non-linear separable data, and classifier is used called as Non-linear SVM classifier.



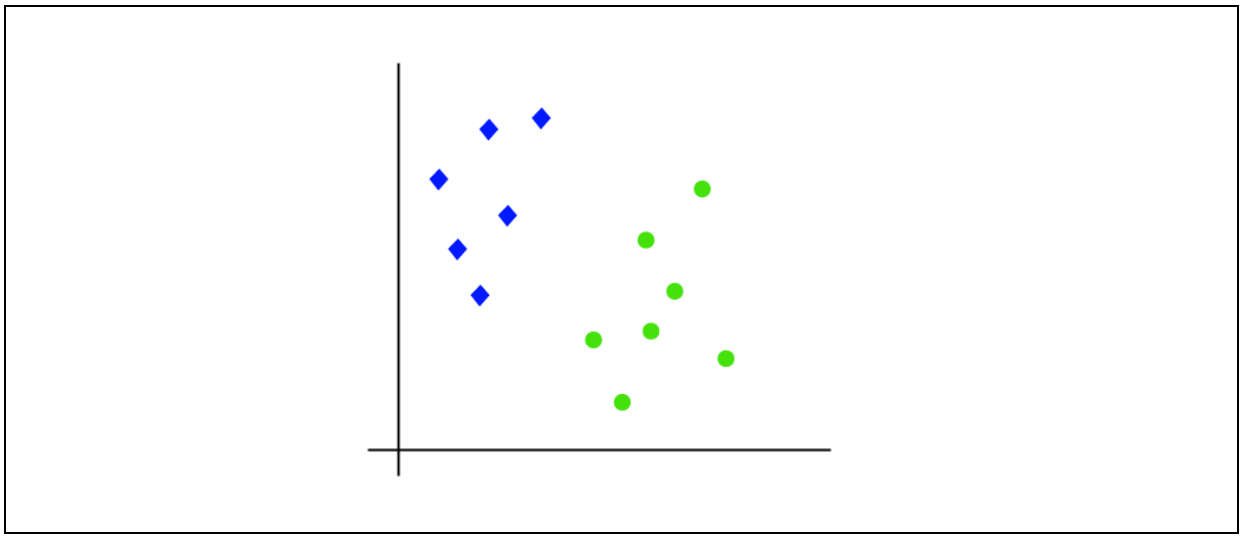

## 8. 2 features forms straight line & 3 features forms plane

- ✓ The dimensions of the hyperplane depend on the features present in the dataset,
  - If there are 2 features then hyperplane will be a straight line
  - If there are 3 features then hyperplane will be a 2-dimension plane

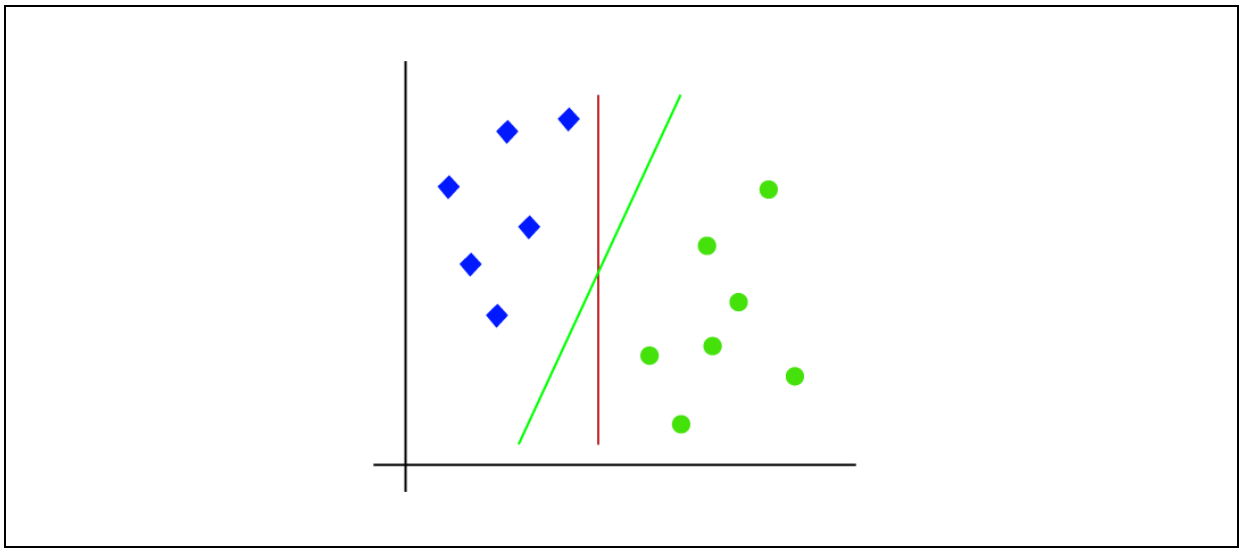
## 9. How does SVM works?

### Linear SVM

- ✓ Assuming that we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2.
- ✓ We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue.

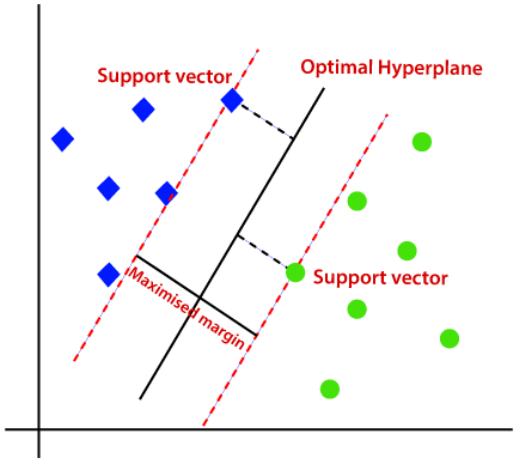


- ✓ Though it is 2-d space we can use straight line to separate these classes
- ✓ There can be multiple lines that can separate these classes too



## 10. Hyperplane and Support vectors

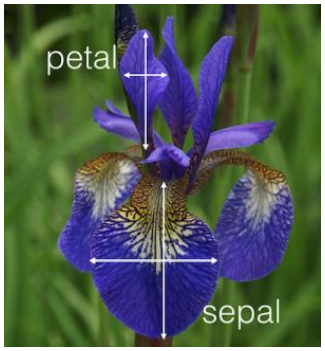
- ✓ SVM algorithm helps to find the best line or decision boundary, this best boundary or region is called as a hyperplane.
- ✓ SVM algorithm finds the closest point of the lines from both the classes; these points are called support vectors.
- ✓ The distance between the vectors and the hyperplane is called as margin.
- ✓ The goal of SVM is to maximize this margin.
- ✓ The hyperplane with maximum margin is called the optimal hyperplane.

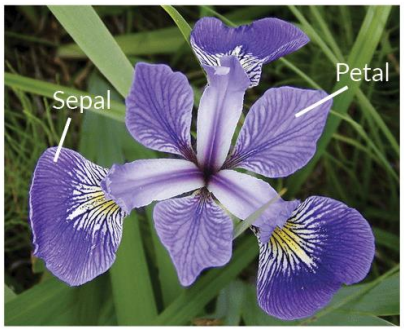


## 11. Use case

- ✓ Assuming that Abhi had a hobby which is interested in distinguishing the species of some iris flowers that he has found
- ✓ He has collected some measurements associated with each iris, which are:
  - The length and width of the petals
  - The length and width of the sepals, all measured in centimetres.

- ✓ She also has the measurements of some irises that have been previously identified to the species
  - setosa,
  - versicolor
  - virginica
- ✓ The goal is to create a machine learning model that can learn from the measurements of these irises whose species are already known.
- ✓ So that we can predict the species for the new irises that she has found.




**Iris Versicolor**

**Iris Setosa**

**Iris Virginica**

## Flower codes

- ✓ Setosa - 0
- ✓ Versicolor - 1
- ✓ Virginica - 2

Program Name: Loading iris dataset
demo1.py

```
from sklearn.datasets import load_iris

iris = load_iris()

print(dir(iris))
```

Output

```
['DESCR', 'data', 'feature_names', 'filename', 'frame', 'target',
'target_names']
```

Program Name: Displaying feature names
demo2.py

```
from sklearn.datasets import load_iris

iris = load_iris()

print(iris.feature_names)
```

Output

```
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal
width (cm)']
```

Program Name: Displaying target names
demo3.py

```
from sklearn.datasets import load_iris

iris = load_iris()

print(iris.target_names)
```

Output

```
['setosa' 'versicolor' 'virginica']
```

Program Name: Displaying data
demo4.py

```
from sklearn.datasets import load_iris

iris = load_iris()

print(iris.data)
```

Output

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
```

Program Name: Length of the data
demo5.py

```
from sklearn.datasets import load_iris

iris = load_iris()

print(len(iris.data))
```

Output

```
150
```

Program Name: Create a Dataframe by using data and features
demo6.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

print(df)
```

Output

```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
..                 ...               ...                ...               ...
145                6.7               3.0                5.2               2.3
146                6.3               2.5                5.0               1.9
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3
149                5.9               3.0                5.1               1.8

[150 rows x 4 columns]
```

Program Name: Adding target column to the dataframe
demo7.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(df.head())
```

Output

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | 0 |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | 0 |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | 0 |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | 0 |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | 0 |

Program Name: Displaying target == 0 flowers
demo8.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(df[df.target==0].head())
```

Output

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  target
0                5.1               3.5                1.4               0.2       0
1                4.9               3.0                1.4               0.2       0
2                4.7               3.2                1.3               0.2       0
3                4.6               3.1                1.5               0.2       0
4                5.0               3.6                1.4               0.2       0
```

Program Name: Displaying length of the target == 0 flowers
demo9.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(len(df[df.target==0]))
```

Output

```
50
```

Program Name: Displaying target == 1 flowers
demo10.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(df[df.target==1].head())
```

Output

```
    sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  target
50                7.0               3.2                4.7               1.4       1
51                6.4               3.2                4.5               1.5       1
52                6.9               3.1                4.9               1.5       1
53                5.5               2.3                4.0               1.3       1
54                6.5               2.8                4.6               1.5       1
```

Program Name: Displaying length of the target == 0 flowers
demo11.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(len(df[df.target==1]))
```

Output

```
50
```

Program Name: Displaying target == 2 flowers
demo12.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(df[df.target==2].head())
```

Output

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target |
|---|---|---|---|---|---|
| 100 | 6.3 | 3.3 | 6.0 | 2.5 | 2 |
| 101 | 5.8 | 2.7 | 5.1 | 1.9 | 2 |
| 102 | 7.1 | 3.0 | 5.9 | 2.1 | 2 |
| 103 | 6.3 | 2.9 | 5.6 | 1.8 | 2 |
| 104 | 6.5 | 3.0 | 5.8 | 2.2 | 2 |

Program Name: Displaying length of the target == 2 flowers
demo13.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

print(len(df[df.target==2]))
```

Output

```
50
```

Program Name: Displaying the flower names
demo14.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target


df['flower_name'] =df.target.apply(lambda x: iris.target_names[x])
print(df)
```

Output

```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  target flower_name
0                  5.1               3.5                1.4               0.2       0      setosa
1                  4.9               3.0                1.4               0.2       0      setosa
2                  4.7               3.2                1.3               0.2       0      setosa
3                  4.6               3.1                1.5               0.2       0      setosa
4                  5.0               3.6                1.4               0.2       0      setosa
..                 ...               ...                ...               ...     ...         ...
145                6.7               3.0                5.2               2.3       2   virginica
146                6.3               2.5                5.0               1.9       2   virginica
147                6.5               3.0                5.2               2.0       2   virginica
148                6.2               3.4                5.4               2.3       2   virginica
149                5.9               3.0                5.1               1.8       2   virginica

[150 rows x 6 columns]
```

Program Name: All setosa flowers
demo15.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

setosa_50 = df[:50]
print(setosa_50.head())
```

Output

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  target flower_name
0                5.1               3.5                1.4               0.2       0      setosa
1                4.9               3.0                1.4               0.2       0      setosa
2                4.7               3.2                1.3               0.2       0      setosa
3                4.6               3.1                1.5               0.2       0      setosa
4                5.0               3.6                1.4               0.2       0      setosa
```

Program Name: All versicolor flowers
demo16.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

versicolor_50 = df[50:100]
print(versicolor_50.head())
```

Output

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target | flower_name |
|---|---|---|---|---|---|---|
| 50 | 7.0 | 3.2 | 4.7 | 1.4 | 1 | versicolor |
| 51 | 6.4 | 3.2 | 4.5 | 1.5 | 1 | versicolor |
| 52 | 6.9 | 3.1 | 4.9 | 1.5 | 1 | versicolor |
| 53 | 5.5 | 2.3 | 4.0 | 1.3 | 1 | versicolor |
| 54 | 6.5 | 2.8 | 4.6 | 1.5 | 1 | versicolor |

Program Name: All virginica flowers
demo17.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

verginica_50 = df[100:]
print(verginica_50.head())
```

Output

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | target | flower_name |
|---|---|---|---|---|---|---|
| 100 | 6.3 | 3.3 | 6.0 | 2.5 | 2 | virginica |
| 101 | 5.8 | 2.7 | 5.1 | 1.9 | 2 | virginica |
| 102 | 7.1 | 3.0 | 5.9 | 2.1 | 2 | virginica |
| 103 | 6.3 | 2.9 | 5.6 | 1.8 | 2 | virginica |
| 104 | 6.5 | 3.0 | 5.8 | 2.2 | 2 | virginica |

Program Name: All types of flowers
demo18.py

```
import pandas as pd
from sklearn.datasets import load_iris

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

df0 = df[:50]
df1 = df[50:100]
df2 = df[100:]

print("All types of flowers")
```

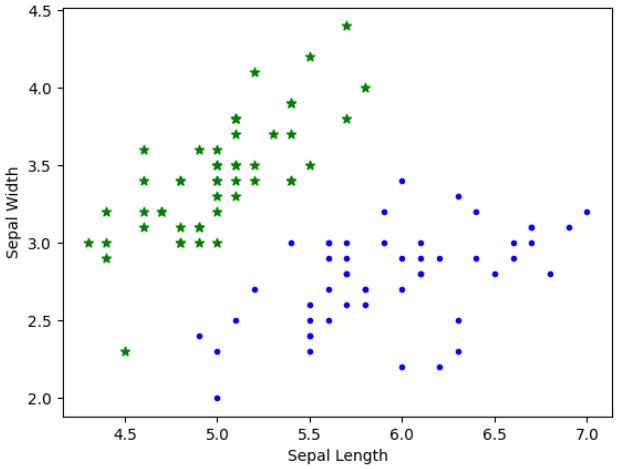
Output

```
All types of flowers
```

Program Name: Plotting
demo19.py

```
import pandas as pd
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

df0 = df[:50]
df1 = df[50:100]
df2 = df[100:]


# Sepal length vs Sepal Width (Setosa vs Versicolor)

plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')

plt.scatter(df0['sepal length (cm)'], df0['sepal width (cm)'],
color="green", marker='*')

plt.scatter(df1['sepal length (cm)'], df1['sepal width (cm)'],
color="blue", marker='.')

plt.show()
```

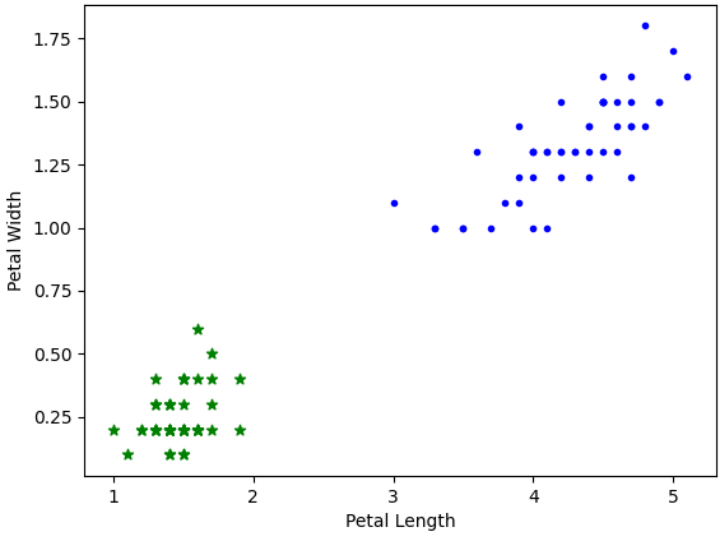
Output



Program Name: Plotting
demo20.py

```
import pandas as pd
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

df0 = df[:50]
df1 = df[50:100]
df2 = df[100:]

# Petal length vs Petal Width (Setosa vs Versicolor)

plt.xlabel('Petal Length')
plt.ylabel('Petal Width')
plt.scatter(df0['petal length (cm)'], df0['petal width (cm)'],
color="green", marker='*')
plt.scatter(df1['petal length (cm)'], df1['petal width (cm)'],
color="blue", marker='.')

plt.show()
```

Output



Program Name: Splitting the data
demo21.py

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

X = df.drop(['target', 'flower_name'], axis='columns')
y = df.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2)

print("Splitting the data")
```

Output

```
Splitting the data
```

Program Name: Model training
demo22.py

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

X = df.drop(['target', 'flower_name'], axis='columns')
y = df.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2)

# Train Using Support Vector Machine (SVM)

model = SVC()
model.fit(X_train, y_train)
print("Model got trained")
```

Output

```
Model got trained
```

Program Name: Model score
demo23.py

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

X = df.drop(['target', 'flower_name'], axis='columns')
y = df.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2)

# Train Using Support Vector Machine (SVM)

model = SVC()
model.fit(X_train, y_train)

print(model.score(X_test, y_test))
```

Output

```
0.9666666666666667
```

Program Name: Model prediction
demo24.py

```
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

iris = load_iris()

df = pd.DataFrame(iris.data, columns=iris.feature_names)

df['target'] = iris.target
df['flower_name'] = df.target.apply(lambda x:
iris.target_names[x])

X = df.drop(['target', 'flower_name'], axis='columns')
y = df.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2)

# Train Using Support Vector Machine (SVM)

model = SVC()
model.fit(X_train, y_train)

print(model.predict([[4.8,3.0,1.5,0.3]]))
```

Output

```
[0]
```