In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\feature_extraction\image.p
y:167: DeprecationWarning: `np.int` is a deprecated alias for the builtin
`int`. To silence this warning, use `int` by itself. Doing this will not m
odify any behavior and is safe. When replacing `np.int`, you may wish to u
se e.g. `np.int64` or `np.int32` to specify the precision. If you wish to
review your current use, check the release note link for additional inform
ation.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  dtype=np.int):
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:30: DeprecationWarning: `np.float` is a deprecated alias for the builtin
`float`. To silence this warning, use `float` by itself. Doing this will n
ot modify any behavior and is safe. If you specifically wanted the numpy s
calar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  method='lar', copy_X=True, eps=np.finfo(np.float).eps,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:167: DeprecationWarning: `np.float` is a deprecated alias for the builti
n `float`. To silence this warning, use `float` by itself. Doing this will
not modify any behavior and is safe. If you specifically wanted the numpy
scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  method='lar', copy_X=True, eps=np.finfo(np.float).eps,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:284: DeprecationWarning: `np.float` is a deprecated alias for the builti
n `float`. To silence this warning, use `float` by itself. Doing this will
not modify any behavior and is safe. If you specifically wanted the numpy
scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_Gram=True, verbose=0,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:862: DeprecationWarning: `np.float` is a deprecated alias for the builti
n `float`. To silence this warning, use `float` by itself. Doing this will
not modify any behavior and is safe. If you specifically wanted the numpy
scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_X=True, fit_path=True,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1101: DeprecationWarning: `np.float` is a deprecated alias for the built
in `float`. To silence this warning, use `float` by itself. Doing this wil
l not modify any behavior and is safe. If you specifically wanted the nump
y scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_X=True, fit_path=True,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1127: DeprecationWarning: `np.float` is a deprecated alias for the built
in `float`. To silence this warning, use `float` by itself. Doing this wil
l not modify any behavior and is safe. If you specifically wanted the nump
y scalar type, use `np.float64` here.
```

```
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, positive=False):
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1362: DeprecationWarning: `np.float` is a deprecated alias for the built
in `float`. To silence this warning, use `float` by itself. Doing this wil
l not modify any behavior and is safe. If you specifically wanted the nump
y scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  max_n_alphas=1000, n_jobs=None, eps=np.finfo(np.float).eps,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1602: DeprecationWarning: `np.float` is a deprecated alias for the built
in `float`. To silence this warning, use `float` by itself. Doing this wil
l not modify any behavior and is safe. If you specifically wanted the nump
y scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  max_n_alphas=1000, n_jobs=None, eps=np.finfo(np.float).eps,
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\least_angle.p
y:1738: DeprecationWarning: `np.float` is a deprecated alias for the built
in `float`. To silence this warning, use `float` by itself. Doing this wil
l not modify any behavior and is safe. If you specifically wanted the nump
y scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.or
g/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/devdoc
s/release/1.20.0-notes.html#deprecations)
  eps=np.finfo(np.float).eps, copy_X=True, positive=False):
```

In [2]:

```
iris=load_iris()
```

```
mation.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.o
rg/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/dev
docs/release/1.20.0-notes.html#deprecations)
  target[i] = np.asarray(ir[-1], dtype=np.int)
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\datasets\base.py:245: De
precationWarning: `np.int` is a deprecated alias for the builtin `int`.
To silence this warning, use `int` by itself. Doing this will not modify
any behavior and is safe. When replacing `np.int`, you may wish to use
e.g. `np.int64` or `np.int32` to specify the precision. If you wish to r
eview your current use, check the release note link for additional infor
mation.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.o
rg/devdocs/release/1.20.0-notes.html#deprecations (https://numpy.org/dev
docs/release/1.20.0-notes.html#deprecations)
  target[i] = np.asarray(ir[-1], dtype=np.int)
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\datasets\base.py:245: De
precationWarning: `np.int` is a deprecated alias for the builtin `int`.
To silence this warning, use `int` by itself. Doing this will not modify
any behavior and is safe. When replacing `np.int`, you may wish to use
```

In [3]:

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

Out[3]:

	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 × 4 columns

In [4]:

```
df['target']=iris.target
df
```

Out[4]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

In [5]:

```
df.info
```

Out[5]:

```
<bound method DataFrame.info of      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

     target
0         0
1         0
2         0
3         0
4         0
..      ...
145       2
146       2
147       2
148       2
149       2

[150 rows x 5 columns]>
```

In [6]:

```
df.isna().sum()
```

Out[6]:

```
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
target               0
dtype: int64
```

In [7]:

```
x=df.drop('target',axis=1)
x.head()
```

Out[7]:

	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

In [8]:

```
y=df.target
y
```

Out[8]:

```
0      0
1      0
2      0
3      0
4      0
      ..
145    2
146    2
147    2
148    2
149    2
Name: target, Length: 150, dtype: int32
```

In [9]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.7)
```

In [10]:

```
logre=LogisticRegression()
logre
```

Out[10]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=Tru
e,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=
0,
                   warm_start=False)
```

In [11]:

```
logre.fit(x_train,y_train)
```

```
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Spec
ify a solver to silence this warning.
  FutureWarning)
C:\Users\hp\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
69: FutureWarning: Default multi_class will be changed to 'auto' in 0.22.
Specify the multi_class option to silence this warning.
  "this warning.", FutureWarning)
```

Out[11]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=Tru
e,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=
0,
                   warm_start=False)
```

In [12]:

```
logaccu=logre.score(x_test,y_test)
print("Accuracy of model:{0}%".format(logaccu*100))
```

```
Accuracy of model:95.55555555555556%
```

In [15]:

```
values=[[5.0,3.6,1.4,0.2]]

result=logre.predict(value)
if result[0]==0:
    print('The species is Setosa')
elif result[0]==1:
    print('The species is Versicolr')
else:
    print('The species is Virginica')
```

```
The species is Setosa
```

In [ ]:

In [ ]: