In [1]:

```
from sklearn.datasets import load_iris
```

In [2]:

```
iris=load_iris()
```

In [3]:

```
X=iris.data
y=iris.target
```

In [4]:

```
from sklearn.linear_model import LogisticRegression
```

In [5]:

```
model=LogisticRegression(multi_class='ovr')
model.fit(X,y)
```

Out[5]:

```
▾         LogisticRegression
LogisticRegression(multi_class='ovr')
```

In [6]:

```
sample=[1.5,.8,3.5,.9]
print(model.predict_proba([sample]))
print(model.predict([sample]))
```

```
[[0.08289626 0.87528925 0.0418145 ]]
[1]
```

In [7]:

```
model.coef_
```

Out[7]:

```
array([[-0.44501376,  0.89999242, -2.32353827, -0.97345836],
       [-0.1792787 , -2.12866718,  0.69665417, -1.27480129],
       [-0.39444787, -0.5133412 ,  2.93087523,  2.41709879]])
```

In [8]:

```
model.intercept_
```

Out[8]:

```
array([  6.69040651,   5.58615272, -14.43121671])
```

In [10]:

```
z0=-0.44501376*1.5+0.89999242*.8+-2.32353827*3.5+-0.97345836*.9+6.69040651
z1=-0.1792787*1.5+-2.12866718*.8+0.69665417*3.5+-1.27480129*.9+5.58615272
z2=-0.39444787*1.5+-0.5133412*.8+2.93087523*3.5+2.41709879*.9+-14.43121671
print(z0,z1,z2)
```

```
-2.2656166629999985 4.90526936 -3.0001092590000003
```

In [11]:

```
model.decision_function([sample])
```

Out[11]:

```
array([[-2.26561665,  4.90526936, -3.00010925]])
```

In [17]:

```
import numpy as np
p0=1/(1+np.exp(-z0))
p1=1/(1+np.exp(-z1))
p2=1/(1+np.exp(-z2))

#normalize probs
pn0=p0/(p0+p1+p2)
pn1=p1/(p0+p1+p2)
pn2=p2/(p0+p1+p2)
print(pn0,pn1,pn2)
```

```
0.08289625540169702 0.875289246767024 0.041814497831278885
```

In [13]:

```
model.predict_proba([sample])
```

Out[13]:

```
array([[0.08289626, 0.87528925, 0.0418145 ]])
```

In [20]:
```
model=LogisticRegression(multi_class='multinomial',max_iter=200)
model.fit(X,y)
```

Out[20]:
```
▾ LogisticRegression
LogisticRegression(max_iter=200, multi_class='multinomial')
```

In [21]:
```
model.coef_
```

Out[21]:
```
array([[-0.42340889,  0.96722201, -2.51717294, -1.07951336],
       [ 0.53440819, -0.32161354, -0.20651822, -0.94415957],
       [-0.1109993 , -0.64560846,  2.72369116,  2.02367293]])
```

In [22]:
```
model.intercept_
```

Out[22]:
```
array([  9.84977931,   2.23796272, -12.08774203])
```

In [23]:
```
sample=[1.5,.8,3.5,.9]
z0=-0.42340889*1.5+0.96722201*.8+-2.51717294*3.5+-1.07951336*.9+9.84977931
z1=0.53440819*1.5+-0.32161354*.8+-0.20651822*3.5+-0.94415957*.9+2.23796272
z2=-0.1109993*1.5+-0.64560846*.8+2.72369116*3.5+2.02367293*.9+-12.08774203
print(z0,z1,z2)
```

```
0.20677626900000057 1.20972679 -1.4165030509999994
```

In [24]:
```
model.decision_function([sample])
```

Out[24]:
```
array([[ 0.20677627,  1.20972677, -1.41650305]])
```

In [25]:
```
p0=np.exp(z0)/(np.exp(z0)+np.exp(z1)+np.exp(z2))
p1=np.exp(z1)/(np.exp(z0)+np.exp(z1)+np.exp(z2))
p2=np.exp(z2)/(np.exp(z0)+np.exp(z1)+np.exp(z2))
print(p0,p1,p2)
```

```
0.2548702643544101 0.6948563749460374 0.05027336069955249
```

In [26]:
```
model.predict_proba([sample])
```

Out[26]:
```
array([[0.25487027, 0.69485637, 0.05027336]])
```

In [28]:
```
model.predict([sample])
```

Out[28]:
```
array([1])
```

In [29]:
```
import sklearn
```

In [30]:
```
sklearn.__version__
```

Out[30]:
```
'1.3.0'
```

In [31]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [32]:
```
model=DecisionTreeClassifier()
model.fit(X,y)
```

Out[32]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [ ]:
```
```