lab10_singlelayerperceptron

November 15, 2022

[ ]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing
```

[ ]:
```
data=pd.read_csv('C:\Coding\ML_python\machine-learning-lab-main\datasets\iris.
 ↪csv')
data.
 ↪columns=['Sepal_len_cm','Sepal_wid_cm','Petal_len_cm','Petal_wid_cm','Type']
data.head(10)
```

[ ]:
```
   Sepal_len_cm  Sepal_wid_cm  Petal_len_cm  Petal_wid_cm    Type
0           5.1           3.5           1.4           0.2  setosa
1           4.9           3.0           1.4           0.2  setosa
2           4.7           3.2           1.3           0.2  setosa
3           4.6           3.1           1.5           0.2  setosa
4           5.0           3.6           1.4           0.2  setosa
5           5.4           3.9           1.7           0.4  setosa
6           4.6           3.4           1.4           0.3  setosa
7           5.0           3.4           1.5           0.2  setosa
8           4.4           2.9           1.4           0.2  setosa
9           4.9           3.1           1.5           0.1  setosa
```

[ ]:
```
def activation_func(value):     #Tangent Hypotenuse\n"
  #return (1/(1+np.exp(-value)))\n",
  return ((np.exp(value)-np.exp(-value))/(np.exp(value)+np.exp(-value)))
```

[ ]:
```
def perceptron_train(in_data,labels,alpha):
  X=np.array(in_data)
  y=np.array(labels)
  weights=np.random.random(X.shape[1])
  original=weights
  bias=np.random.random_sample()
  for key in range(X.shape[0]):
    a=activation_func(np.matmul(np.transpose(weights),X[key]))
    yn=0
    if a>=0.7:
      yn=1
    elif a<=(-0.7):
      yn=-1
    weights=weights+alpha*(yn-y[key])*X[key]
  print('Iteration '+str(key)+': '+str(weights))
  print('Difference: '+str(weights-original))
  return weights
```

[ ]:
```
def perceptron_test(in_data,label_shape,weights):
  X=np.array(in_data)
  y=np.zeros(label_shape)
  for key in range(X.shape[1]):
    a=activation_func((weights*X[key]).sum())
    y[key]=0
    if a>=0.7:
      y[key]=1
    elif a<=(-0.7):
      y[key]=-1
  return y
```

[ ]:
```
def score(result,labels):
  difference=result-np.array(labels)
  correct_ctr=0
  for elem in range(difference.shape[0]):
    if difference[elem]==0:
      correct_ctr+=1
  score=correct_ctr*100/difference.size
  print('Score='+str(score))
```

[ ]:
```
divider = np.random.rand(len(data)) < 0.70
d_train=data[divider]
d_test=data[~divider]
```

[ ]:
```
d_train_y=preprocessing.LabelEncoder().fit_transform(d_train['Type'])
d_train_X=d_train.drop(['Type'],axis=1)
# Dividing d_train into data and labels/targets\n",
d_test_y=preprocessing.LabelEncoder().fit_transform(d_test['Type'])
d_test_X=d_test.drop(['Type'],axis=1)
```

[ ]:
```
# Learning rate\n",
alpha = 0.01
# Train\n",
weights = perceptron_train(d_train_X, d_train_y, alpha)
```

```
Iteration 113: [-0.92846459  0.04778317 -1.43781047 -0.11035329]
Difference: [-0.961 -0.013 -1.866 -0.831]
```

[ ]:
```
result_test=perceptron_test(d_test_X,d_test_y.shape,weights)
```

[ ]:
```
score(result_test,d_test_y)
```

```
Score=19.444444444444443
```