4. Perceptron

March 28, 2022

[1]:
```
# Import Libraries
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
```

[2]:
```
# Load dataset
data = load_iris()

# Get features and target
X=data.data
y=data.target

nlabels = len(set(y))

print(X.shape)
```

```
(150, 4)
```

[3]:
```
# Get dummy variable
y = pd.get_dummies(y).values
```

[4]:
```
#Split data into train and test data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1)
```

[5]:
```
# Initialize variables
learning_rate = 0.1
iterations = 5000
N = y_train.size

# number of input features
input_size = X_train.shape[1]

# number of hidden layers neurons
hidden_size = 2

# number of neurons at the output layer
output_size = nlabels
```

[6]:
```
np.random.seed(10)

# initializing weight for the hidden layer
W1 = np.random.normal(scale=0.5, size=(input_size, hidden_size))

# initializing weight for the output layer
W2 = np.random.normal(scale=0.5, size=(hidden_size , output_size))
```

[7]:
```
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
```

[8]:
```
for itr in range(iterations):

    # feedforward propagation on hidden layer
    Z1 = np.dot(X_train, W1)
    A1 = sigmoid(Z1)         # output of hidden neurons

    # on output layer
    Z2 = np.dot(A1, W2)
    A2 = sigmoid(Z2)         # output of output neurons

    # backpropagation
    E1 = A2 - y_train
    dW1 = E1 * A2 * (1 - A2)        # (p-t)p(1-p)

    E2 = np.dot(dW1, W2.T)         # (p-t)p(1-p)w_jk
    dW2 = E2 * A1 * (1 - A1)       # [[(p-t)p(1-p)]w_jk]h(1-h)


    # weight updates
    # [(p-t)p(1-p)]h_j
    W2_update = np.dot(A1.T, dW1) / N
    # [(p-t)p(1-p)(w_jk)h(1-h)]x_i
    W1_update = np.dot(X_train.T, dW2) / N

    W2 = W2 - learning_rate * W2_update
    W1 = W1 - learning_rate * W1_update
```

[9]:
```
# feedforward
Z1 = np.dot(X_test, W1)
A1 = sigmoid(Z1)

Z2 = np.dot(A1, W2)
A2 = sigmoid(Z2)

y_pred = A2
acc = y_pred.argmax(axis=1) == y_test.argmax(axis=1)
```

```
print("Accuracy=",acc.mean())
```

```
Accuracy= 0.6
```

[ ]: