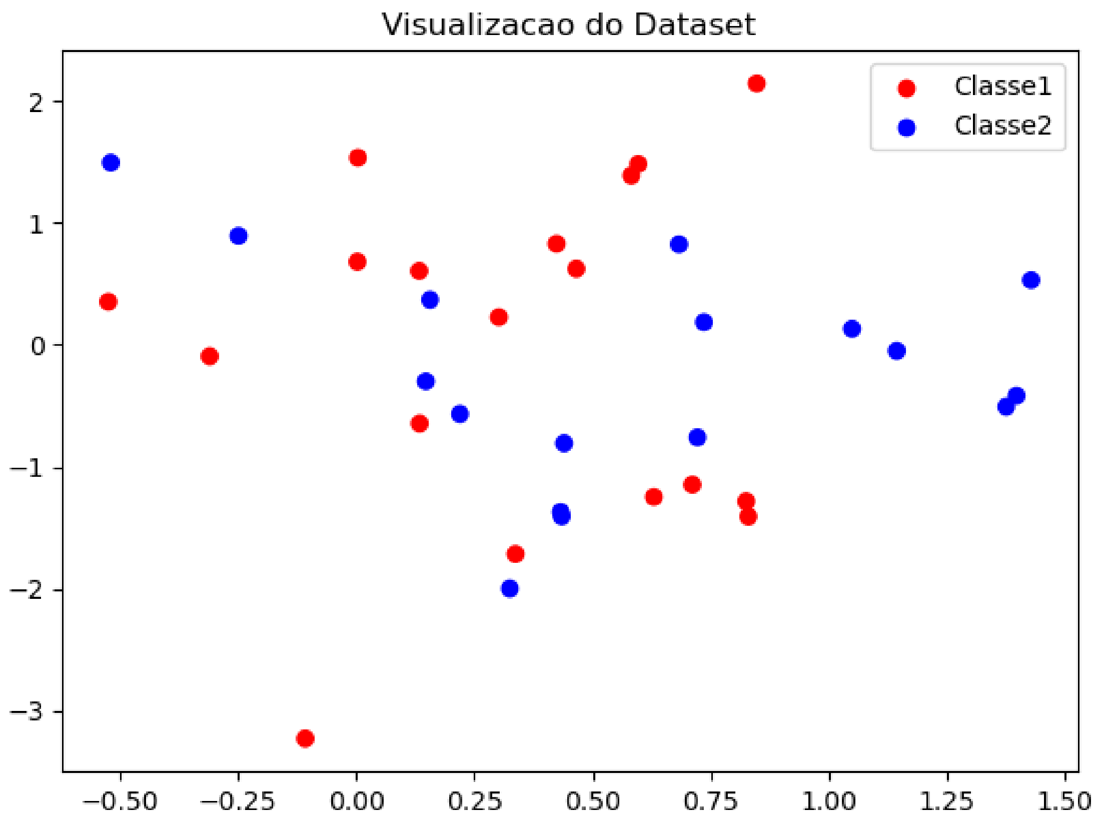
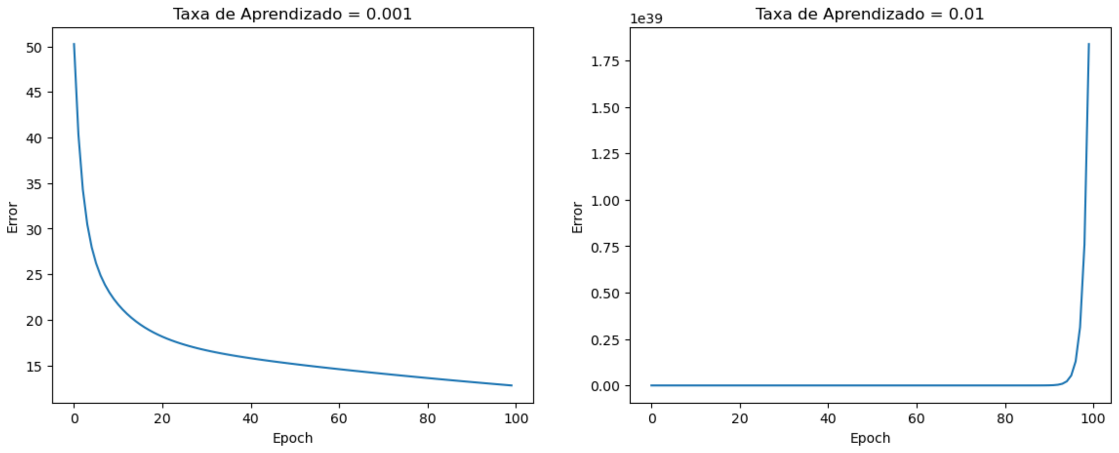
RedeAdaline

September 28, 2022

[1]:
```python
import matplotlib.pyplot as plt #Para visualizacao dos dados e do erro
import numpy as np #Biblioteca de manipulacao de arrays Numpy
from matplotlib.colors import ListedColormap #Lista de cores para plotagens
import pandas as pd



### Carregar iris dataset
df = pd.read_csv('Dados_Treinamento_Sinal.csv',header=None)
df.head()

X = df.iloc[0:35,[0,1,2,3]].values
y = df.iloc[0:35,4].values

#print(X)
#print("Y:", y)

### Assumindo que Setosa(0) seja -1 e Versicolor = 1

### Plotar o grafico
### vermelhos ----> Classe2 (-1)
### azuis ----> Classe1 (1)

cm_bright = ListedColormap(['#FF0000', '#0000FF'])
plt.figure(figsize=(7,5))
plt.scatter(X[:,0], X[:,1], c=y, cmap=cm_bright)
plt.scatter(None, None, color = 'r', label='Classe1')
plt.scatter(None, None, color = 'b', label='Classe2')
plt.legend()
plt.title('Visualizacao do Dataset')
plt.show()

###Construindo Adaline
class Adaline(object):
    def __init__(self, eta = 0.001, epoch = 100):
        self.eta = eta
        self.epoch = epoch
    def fit(self, X, y):
        np.random.seed(16)
        self.weight_ = np.random.uniform(-1, 1, X.shape[1] + 1)
        self.error_ = []

        cost = 0
        for _ in range(self.epoch):

            output = self.activation_function(X)
            error = y - output

            self.weight_[0] += self.eta * sum(error)
            self.weight_[1:] += self.eta * X.T.dot(error)

            cost = 1./2 * sum((error**2))
            self.error_.append(cost)

        return self

    def net_input(self, X):
        """Calculo da entrada z"""
        return np.dot(X, self.weight_[1:]) + self.weight_[0]
    def activation_function(self, X):
        """Calculo da saida da funcao g(z)"""
        return self.net_input(X)
    def predict(self, X):
        """Retornar valores binaros 0 ou 1"""
        return np.where(self.activation_function(X) >= 0.0, 1, -1)


###Plotando erros apos 100 epocas
names = ['Taxa de Aprendizado = 0.001', 'Taxa de Aprendizado = 0.01']
classifiers = [Adaline(), Adaline(eta = 0.01)]
step = 1
plt.figure(figsize=(14,5))
for name, classifier in zip(names, classifiers):
    ax = plt.subplot(1, 2, step)
    clf = classifier.fit(X, y)
    ax.plot(range(len(clf.error_)), clf.error_)
    ax.set_ylabel('Error')
    ax.set_xlabel('Epoch')
    ax.set_title(name)

    step += 1

plt.show()
```

```
### Plotando as fronteiras de decisao com Adaline
clf = Adaline()
clf.fit(X, y)

A = [0.4329,-1.3719,0.7022,-0.8535] # Classe 1
B = [0.3024,0.2286,0.8630,2.7909] #Classe -1

print (clf.predict(A))

print (clf.predict(B))
```





Taxa de Aprendizado = 0.001
Taxa de Aprendizado = 0.01
Error
Epoch
1e39

```
1
-1
```

```
[ ]:
```