```
import numpy as np


class Perceptron:
    def __init__(self, lenght, eta=0.001, tolerance=0.0001, max_epochs=10000):
        self.eta = eta
        self.max_epochs = max_epochs
        self.epoch = 0
        self.tolerance = tolerance
        self.errors_per_epoch = []
        self.lenght = lenght
        self.weights = np.random.rand(lenght + 1)
        self.erro = 1

    def train(self, x, y):
        n = len(x)
        error_tolerance = self.tolerance + 1
        while ((self.epoch < self.max_epochs) and (error_tolerance >= self.toleranc
            acumulated_error = 0
            for index in range(n):
                output = self.sum_function(np.asfarray(x[index]))
                self.erro = y[index] - output
                acumulated_error += (self.erro ** 2)
                # delta rule
                self.weights[1:] += self.eta *  np.array(x[index]).T.dot(self.erro)
                self.weights[0] += self.eta * self.erro
            self.epoch += 1
            self.errors_per_epoch.append(acumulated_error / n)
            error_tolerance = acumulated_error / n
        return self.weights

    def step_function(self, sum):
        if (sum >= 0):
            return 1
        return 0

    def predict(self, x):
        return self.sum_function(x)

    def sum_function(self, X):
        return self.step_function(X.dot(self.weights[1:]) + self.weights[0])




import plotly.graph_objects as go
import plotly.express as px
import pandas as pd
import random
import os
```