МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №1
по курсу «Нейроинформатика»

Персептроны. Процедура обучения Розенблатта.

Выполнил: Д. Д. Син
Группа: 8О-407Б
Преподаватели: Н.П Аносова

Москва, 2021

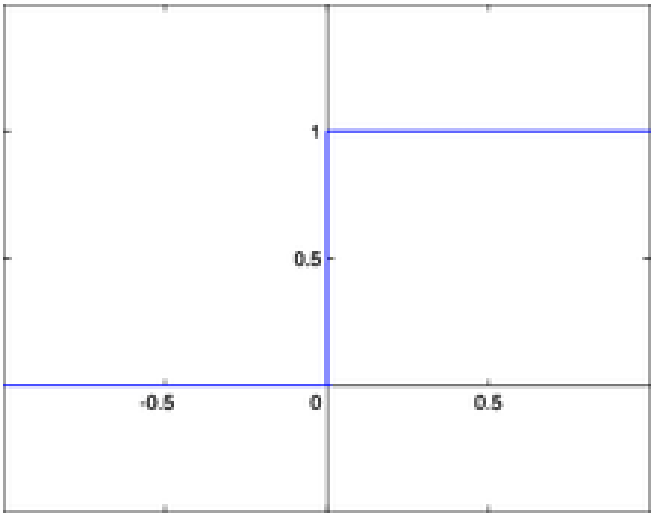
Постановка задачи

Целью работы является исследование свойств персептрона Розенблатта и его применение для решения задачи распознавания образов.

1. Для первой обучающей выборки построить и обучить сеть, которая будет правильно относить точки к двум классам. Отобразить дискриминантную линию и проверить качество обучения.
2. Изменить обучающее множество так, чтобы классы стали линейно неразделимыми. Проверить возможности обучения по правилу Розенблатта.
3. Для второй обучающей выборки построить и обучить сеть, которая будет правильно относить точки к четырем классам. Отобразить дискриминантную линию и проверить качество обучения. Проверить качество, на случайно заданном множестве состоящим из пяти элементов.

Метод решения

Для решения задачи реализуем персептрон Розенблата для двух входов и одного выхода в первом пункте и для двух входов и двух выходов во втором пункте. В качестве функции активации возьмем пороговую функцию



Обучение будем проводить по правилу Розенблата, после чего покажем графики разделяющих прямых.

Описание работы программы

Классификация точек на 2 класса.

Реализация персептрона и обучения с предсказанием

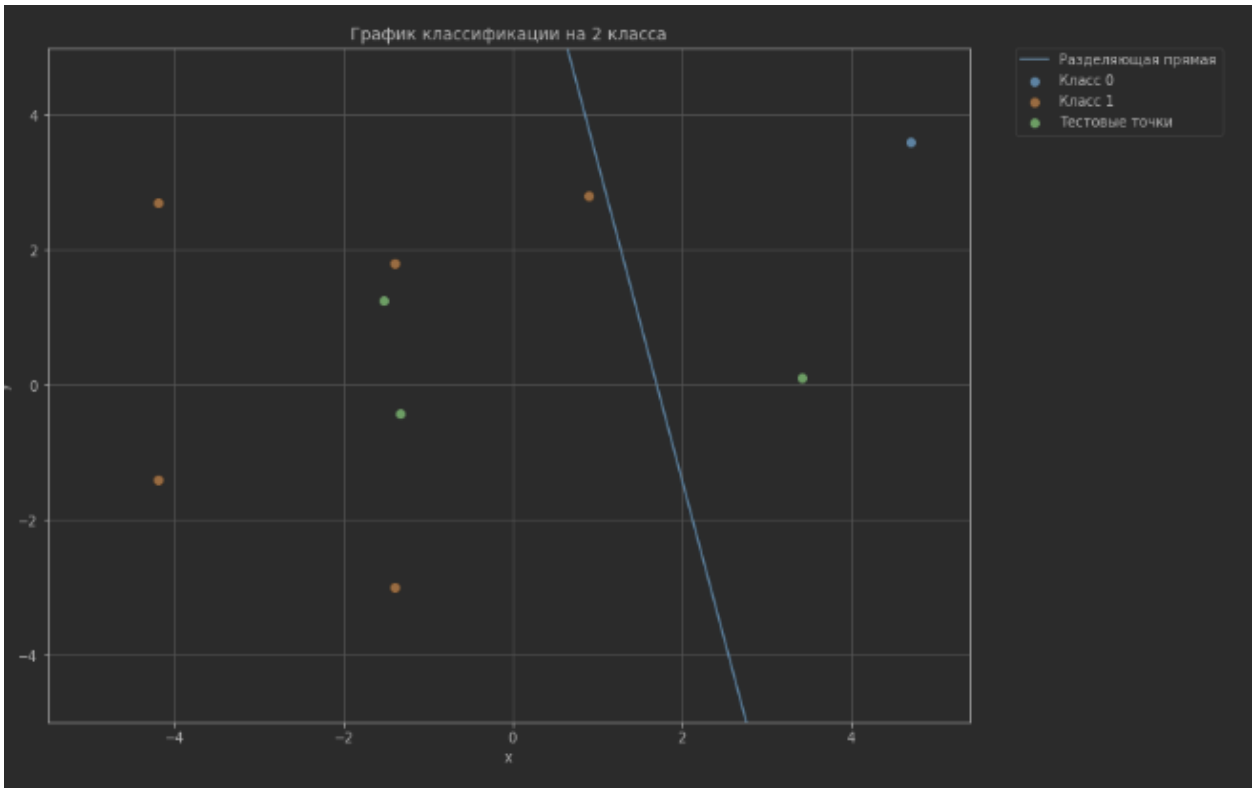
```
class Perceptron:

   def __init__(self, in_size, out_size, learning_rate):
            self.w = np.random.randn(out_size, in_size + 1) /
np.sqrt(in_size)
       self.learning_rate = learning_rate

   def loss(self, x):
       return np.vectorize(lambda t: 1 if t > 0 else 0)(x)

   def fit(self, X, y, epochs=10):
       X = np.c_[X, np.ones((X.shape[0]))]
       for epoch in np.arange(0, epochs):
           for (x, target) in zip(X, y):
               p = self.loss(np.dot(x, self.w.T))
               if not np.array_equal(p, target):
                   error = np.reshape(p - target, (2, 1))
                   self.w -= self.learning_rate * error * x
            print(f'Epoch: {epoch+1}, error: {error.T}, weights:
{self.w}')

   def predict(self, X):
       X = np.atleast_2d(X)
       X = np.c_[X, np.ones((X.shape[0]))]
       return self.loss(np.dot(X, self.w.T))

   def get_weights(self):
       return self.w
```

Обучение на точках из условия:

```
perceptron = Perceptron(2, 0.1)
perceptron.fit(points1, labels1)
```

Вывод программы на 10 эпохах обучения. Получилось 3 веса, так как еще мы используем свободный член b

Epoch: 1, error: -1, weights: [-0.48921511 0.12613385 -0.73345472]
Epoch: 2, error: -1, weights: [-0.53921511 0.10613385 -0.53345472]
Epoch: 3, error: -1, weights: [-0.58921511 0.08613385 -0.33345472]
Epoch: 4, error: -1, weights: [-0.63921511 0.06613385 -0.13345472]
Epoch: 5, error: -1, weights: [-0.68921511 0.04613385 0.06654528]
Epoch: 6, error: -1, weights: [-0.59921511 0.32613385 0.16654528]
Epoch: 7, error: -1, weights: [-0.59921511 0.32613385 0.16654528]
Epoch: 8, error: -1, weights: [-0.59921511 0.32613385 0.16654528]
Epoch: 9, error: -1, weights: [-0.59921511 0.32613385 0.16654528]
Epoch: 10, error: -1, weights: [-0.59921511 0.32613385 0.16654528]



Проверка на сгенерированных точках

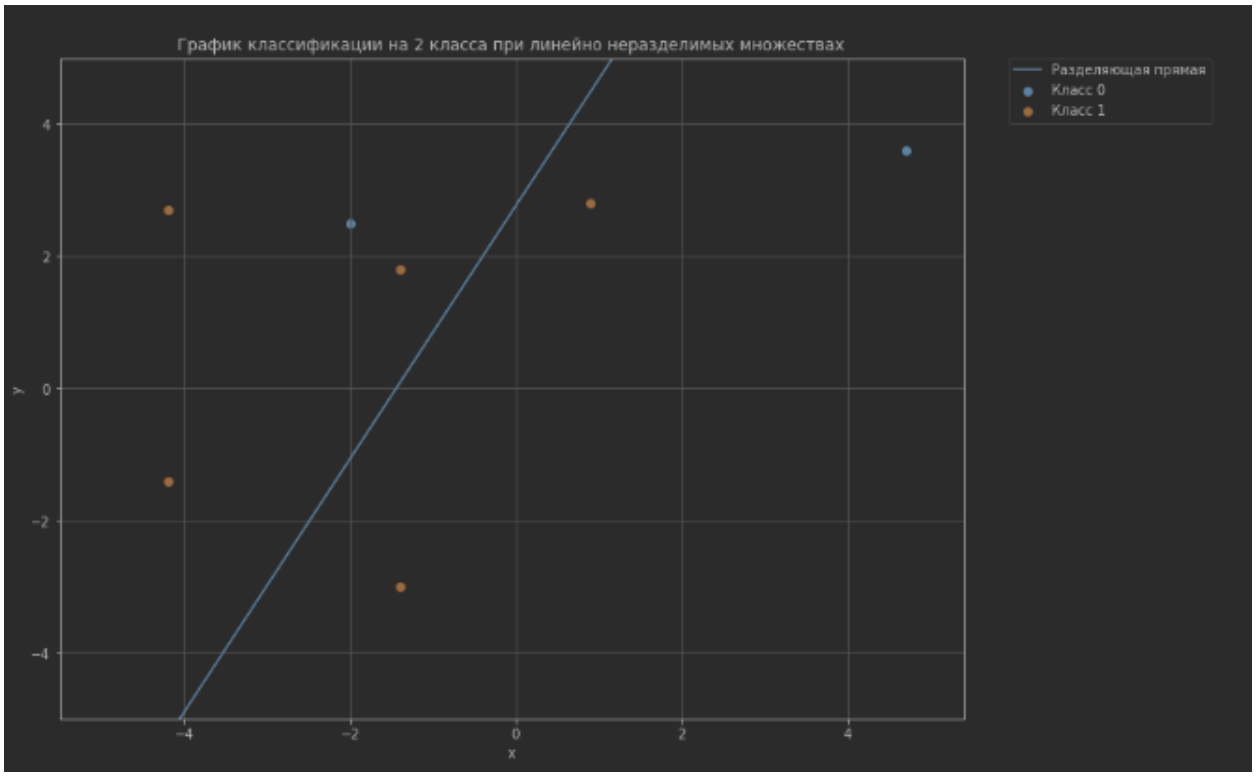
```
for test_point in test_points1:
        print(f'Point:  {test_point}  Predict  class:
{perceptron.predict(test_point)}')
Point: [-1.33500985 -0.42233552] Predict class: 1
Point: [3.4167417  0.11553175] Predict class: 0
Point: [-1.53051761  1.25507008] Predict class: 1
```

Добавление точки для получения двух линейно неразделимых множеств.

При обучении линейно неразделимых множеств точек мы уже не можем хорошо обучить персептрон

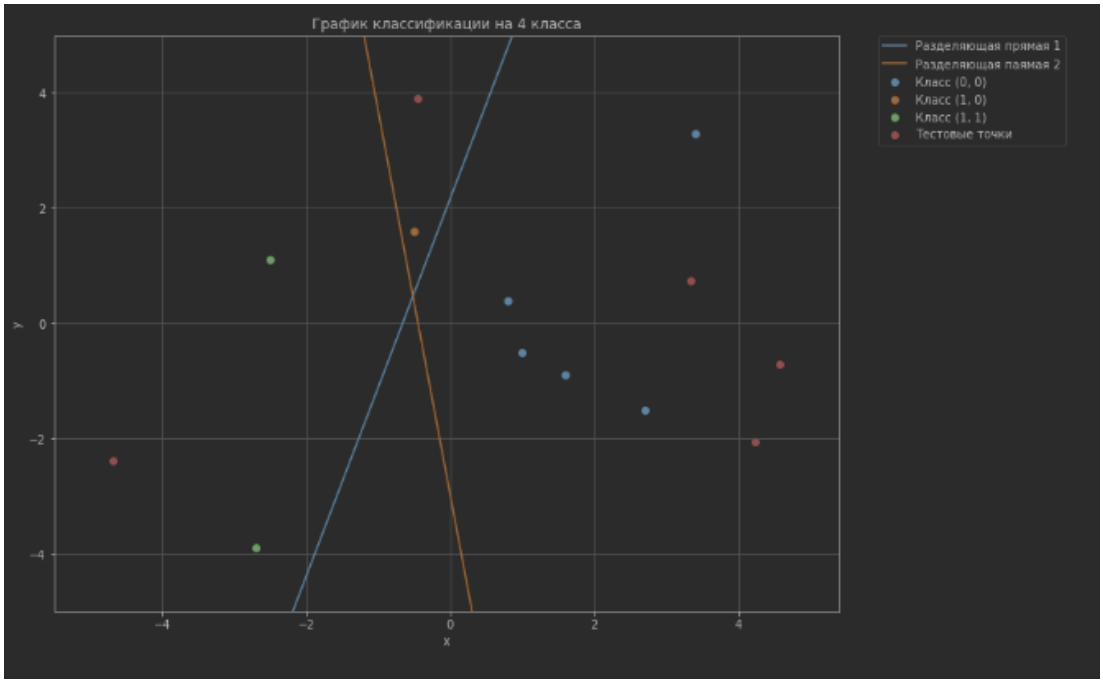
Вывод для 10 эпох

Epoch: 1, error: 1, weights: [-0.69401913 0.15471824 0.2843495]
Epoch: 2, error: 1, weights: [-0.49401913 -0.09528176 0.1843495]
Epoch: 3, error: 1, weights: [-0.20401913 -0.06528176 0.1843495]
Epoch: 4, error: 1, weights: [-0.05401913 -0.33528176 0.2843495]
Epoch: 5, error: 1, weights: [0.09598087 -0.12528176 0.3843495]
Epoch: 6, error: 1, weights: [-0.22401913 -0.27528176 0.3843495]
Epoch: 7, error: 1, weights: [0.06598087 -0.24528176 0.3843495]
Epoch: 8, error: 1, weights: [0.12598087 -0.31528176 0.3843495]
Epoch: 9, error: 1, weights: [0.18598087 -0.38528176 0.3843495]
Epoch: 10, error: 1, weights: [0.33598087 -0.17528176 0.4843495]



Классификация точек на 4 класса.

Epoch: 1, error: [[-1 0]], weights: [[-0.85588567 -0.31851612 -1.09648831]
 [-0.58024899 0.41551593 0.10798158]]
Epoch: 2, error: [[-1 1]], weights: [[-0.90588567 -0.15851612 -0.99648831]
 [-0.53024899 0.25551593 0.00798158]]
Epoch: 3, error: [[-1 1]], weights: [[-0.95588567 0.00148388 -0.89648831]
 [-0.48024899 0.09551593 -0.09201842]]
Epoch: 4, error: [[-1 1]], weights: [[-1.00588567 0.16148388 -0.79648831]
 [-0.43024899 -0.06448407 -0.19201842]]
Epoch: 5, error: [[-1 0]], weights: [[-1.05588567 0.32148388 -0.69648831]
 [-0.43024899 -0.06448407 -0.19201842]]
Epoch: 6, error: [[-1 0]], weights: [[-1.05588567 0.32148388 -0.69648831]
 [-0.43024899 -0.06448407 -0.19201842]]
Epoch: 7, error: [[-1 0]], weights: [[-1.05588567 0.32148388 -0.69648831]
 [-0.43024899 -0.06448407 -0.19201842]]
Epoch: 8, error: [[-1 0]], weights: [[-1.05588567 0.32148388 -0.69648831]
 [-0.43024899 -0.06448407 -0.19201842]]
Epoch: 9, error: [[-1 0]], weights: [[-1.05588567 0.32148388 -0.69648831]
 [-0.43024899 -0.06448407 -0.19201842]]
Epoch: 10, error: [[-1 0]], weights: [[-1.05588567 0.32148388 -0.69648831]
 [-0.43024899 -0.06448407 -0.19201842]]



Выводы

В данной лабораторной работе реализовали персептрон Розенблата на python. Исследовали возможности классификации точен на 2 и 4 класса и проверили, что при линейно неразделимых множествах персептрон не дает хороших результатов. Но уже такая простая модель с взвешенным умножением, которую придумал Розенблат дает хорошие результаты, там где множества заранее линейно разделимы.