**13-я проблема Гильберта.** Можно ли решить общее уравнение седьмой степени с помощью функций, зависящих только от двух переменных? (Можно ли функцию нескольких переменных представитьт в виде суперпозиции нескольких непрерывных функций двух переменных?)

**Теорема Колмогорова — Арнольда.** Многомерная непрерывная функция может быть представлена в виде конечной композиции непрерывных функций одной переменной и операции сложения.

Или: любая непрерывная функция, n вещественных переменных, может быть представлена в виде суммы функций, имеющих своим аргументом суммы непрерывных функций одного аргумента.

**Теорема Цибенко.** Пусть непрерывная сигмоидная функция

тогда конечная сумма

сколь угодно точно приближает любую непрерывную функцию . Другими словами, для сколь угодно малого выполняется условие

Проложим тропинку к нейронным сетям. Для начала в качестве сигмоидной функции используем функцию индикатор

тогда

Нейрон:

или

здесь – входные сигналы;

–набор весов;

*b* – смещение или *h=-b* – порог срабатывания;

Изображение выглядит как диаграмма

Автоматически созданное описание

Что можно получить в простейшем случае одного нейрона с двумя входами?

Для операции одного нейрона недостаточно. Применим формулу:

Используем

Тогда

От одного нейрона мы перешли к суперпозиции нейронов: выход одних нейронов подаём на вход других.

Изображение выглядит как диаграмма

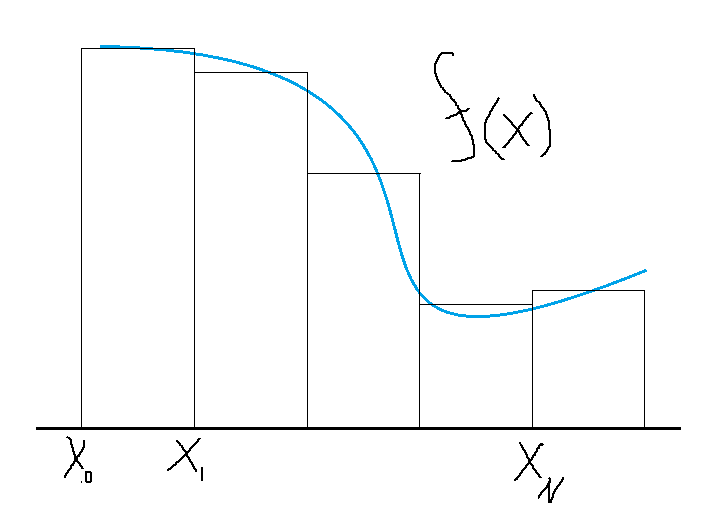
Автоматически созданное описаниеДля индикатора "столбик"

комбинация нейронов имеет вид:

Изображение выглядит как диаграмма

Автоматически созданное описание

Если взять скрытый слой с достаточно большим числом нейронов, то мы можем аппроксимировать любую гладкую функцию.



Выразим индикатор через нейроны:

Изображение выглядит как диаграмма

Автоматически созданное описание

здесь

(здесь *+f* и *-f* для двух рядом стоящих весов)

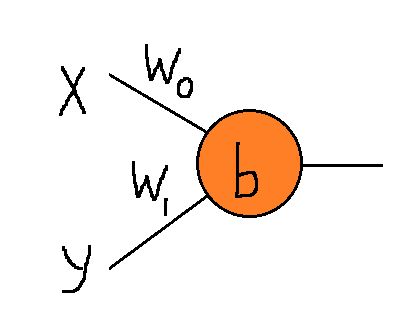
***Логистическая регрессия.***

Основная идея логистической регрессии заключается в том, что пространство исходных значений может быть разделено линейной границей (т. е. прямой) на две соответствующих классам области.

Уравнение

задаёт (n-1)-гиперплоскость в n-мерном пространстве. Пространство разбивается на две области

соответствующие разделению множества объектов на два класса (линейный классификатор).

 Этот классификатор можно реализовать через один нейрон. Ограничимся двумерным случаем:

Значение 0 или 1 на выходе соответствуют принадлежности к одному из двух кластеров.

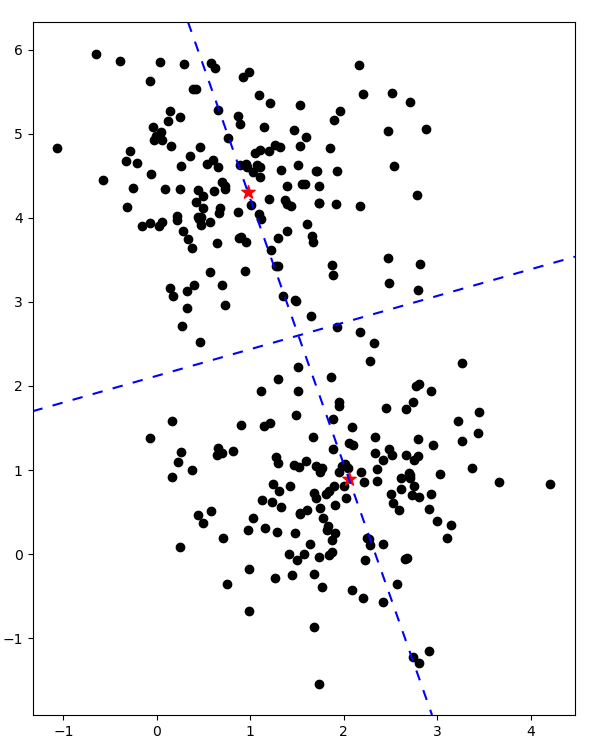
Теперь относительно значений весов и смещения *b*.

Если известны центры двух кластеров , то прямая проходящая через эти центры имеет вид:

Координаты центр соответствующего отрезка равны

Уравнение прямой, проходящей через точку и перпендикулярной прямой проходящей через центры кластеров имеет вид:

Этому уранению следует придать симметричный относительно x-y вид, в этом случае мы избежим неприятностей при .



Если кластеры одинаковы по размеру, то эта прямая будет являться разделителем в линейном классификаторе. В этом случае для весов и смещения *b* получаем следующие значения:

***Программная реализация:***

# Linear classifier

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn import datasets

from sklearn import preprocessing

from math import exp

POINT\_N = 300

DIM\_N = 2

CLUST\_N = 2

data, target, Center = datasets.make\_blobs(n\_samples=POINT\_N, centers=CLUST\_N, cluster\_std=0.8, n\_features=DIM\_N, center\_box=(-10,10), random\_state=0, return\_centers=True)

slope = (Center[1][1]-Center[0][1])/(Center[1][0]-Center[0][0])

C0 = [(Center[0][0]+Center[1][0])/2, (Center[0][1]+Center[1][1])/2]

print("w1=", Center[1][0]-Center[0][0])

print("w2=", Center[1][1]-Center[0][1])

print("b=", -(Center[1][0]-Center[0][0])\*C0[0] - (Center[1][1]-Center[0][1])\*C0[1])

plt.figure(figsize=(7,9))

plt.scatter(data[:,0], data[:,1], c='black', marker='o')

plt.scatter(Center[:,0], Center[:,1], c='red', marker="\*", s=100)

plt.axline((Center[0][0], Center[0][1]), slope=slope, color="blue", linestyle=(0, (5, 5)))

plt.axline((C0[0], C0[1]), slope=-1.0/slope, color="blue", linestyle=(0, (5, 5)))

plt.show()

Изображение выглядит как снимок экрана, диаграмма

Автоматически созданное описание

class NNET0:

def \_\_init\_\_(self):

self.input\_nodes = DIM\_N

self.output\_nodes = 1

self.weights\_input\_to\_output = np.array([[Center[1][0]-Center[0][0]], [Center[1][1]-Center[0][1]]])

self.output\_bias = np.array([ -(Center[1][0]-Center[0][0])\*C0[0] - (Center[1][1]-Center[0][1])\*C0[1] ])

def activation\_function(self, x):

return (1 if x>0 else 0)

def run(self, features):

input\_output = np.dot(features, self.weights\_input\_to\_output)

return self.activation\_function(input\_output+self.output\_bias)

network = NNET0()

pred = []

for i in range(POINT\_N):

pred.append(network.run(data[i]))

plt.figure(figsize=(8,8))

for i in range(POINT\_N):

if pred[i]:

plt.scatter(data[i][0], data[i][1], c='blue', marker='o')

else:

plt.scatter(data[i][0], data[i][1], c='green', marker='o')

plt.scatter(Center[:,0], Center[:,1], c='red', marker="\*", s=100)

plt.axline((C0[0], C0[1]), slope=-1.0/slope, color="black", linestyle=(0, (5, 5)))

plt.show()

print(1.0\*sum(target==pred)/POINT\_N)

print('----------------------------------------------------------')

Изображение выглядит как Красочность, снимок экрана, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.



**Задание:**

Изображение выглядит как диаграмма, линия

Автоматически созданное описание

Построить нейросеть для реализации линейного классификатора в соответствии с приведённым рисунком.