به نام خدا

دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر

شبکه های عصبی و یادگیری عمیق

تمرین سری اول extera

نام و نام خانوادگی	فاطمه حقیقی
شماره دانشجویی	810195385
تاریخ ارسال گزارش	۲۱ اسفند ۱۳۹۸

فهرست گزارش سوالات

سوال 1 – طبقه بندی کاراکترها

پاسخ سوال ۱:

*در این پیاده سازی learning rate = 0.005 در نظر گرفته شده است.

با استفاده از نورون perceptron شبکه ی خواسته شده پیاده سازی شد. حداقل epoch لازم برای train شدن شبکه به وسیله ی train data برابر با ۴ می باشد. در صورتی که تعداد epoch از این مقدار کمتر باشد، شبکه به درستی آموزش داده نمی شود.

پاسخ سوال۲:

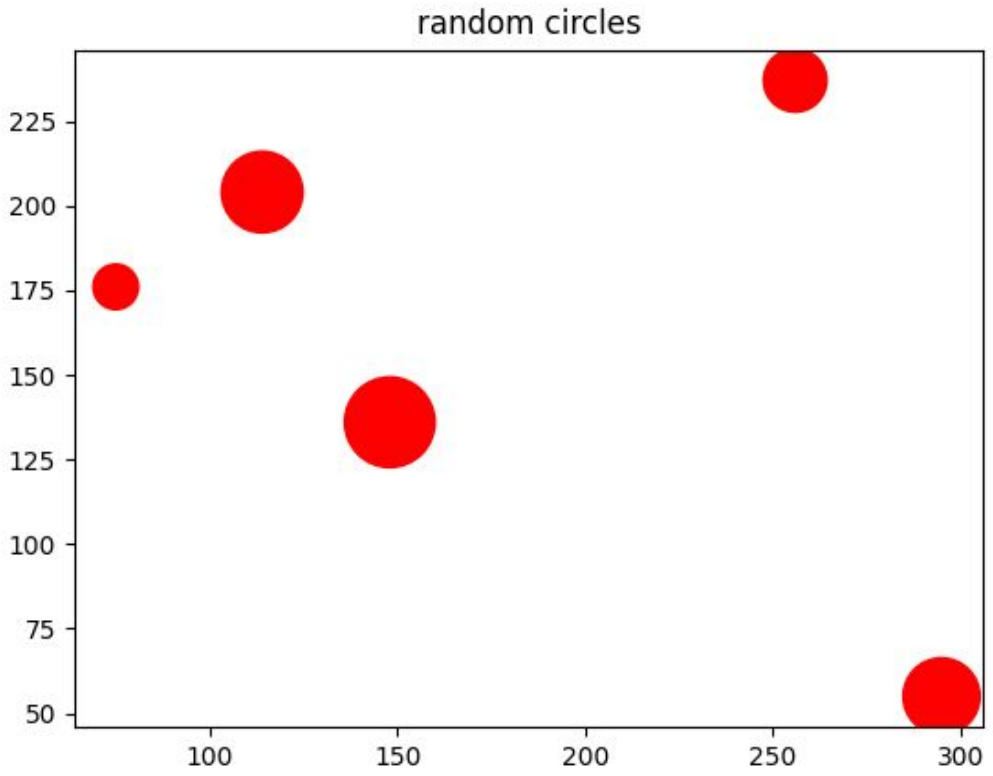
مقدار خطا پس از اجرای test data برروی شبکه ی آموزش دیده به صورت زیر است:

Percent of Error in NN: 0.05714285714285714

سوال ۲ – تغییر فضای ورودی در شبکه ی تک لایه

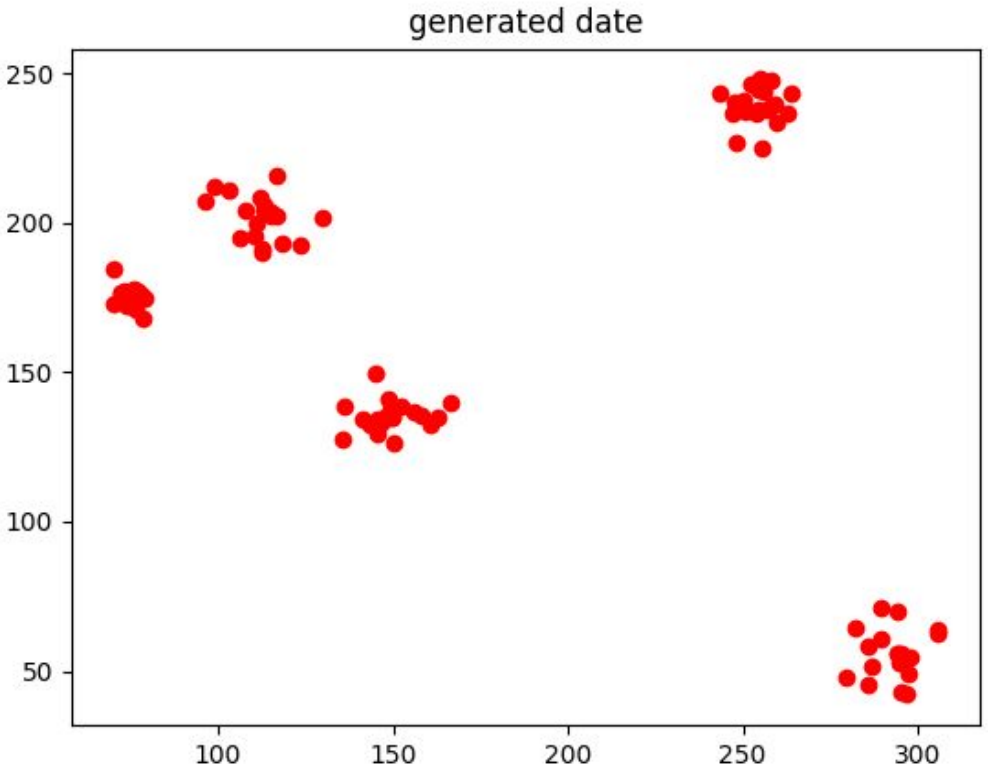
برای حل این سوال ابتدا داده ه های تصادفی را ایجاد کردیم(که در آن ها مقدار xو y مرکز دایره و شعاع آن یک مقدار رندم است). همچنین در آن بررسی کردیم که هر دایره ای که ایجاد می شود، با دایره های دیگر تطابق یا تلاقی نداشته باشد. در صورت وجود این مسئله مجددا مقادیر x و y و r را به طور رندم ایجاد می کنیم.

دایره های ایجاد شده به صورت زیر می باشد:



شکل ۱: دایره ها ی ایجاد شده به صورت رندم

سپس به تعداد ۲۰ نقطه ی تصادفی از هر کدام از دایره ها تولید کردیم. تصویر این نقاط به صورت زیر است:



شکل ۲: داده های نقطه ای انتخاب شده از دایره های تصادفی

پس از آن داده ها را به وسیله ی شبکه ی perceptron که کد آن به صورت زیر است learnکردیم:

```python
class Perceptron_neuron():
    def __init__(self, weights, a, bias, epoch, function):
        self.weights = weights
        self.bias = bias
        self.a = a
        self.out = 0
        self.epoch = epoch
        self.update_counter = 0
        self.function = function

    def update_rule(self, inputs, t):
        self.update_counter += 1
        for i in range(len(inputs)):
            h = self.get_h_value(inputs[i])
            if h - t[i] != 0:
                self.bias = self.bias + self.a * t[i]
                self.weights = self.weights + self.a * inputs[i] *t[i]

    def get_weights(self):
        return self.weights

    def get_bias(self):
        return self.bias

    def get_net_value(self, inputs):
        out = 0
        for i in range(len(inputs)):
            out += self.weights[i] * inputs[i]

        out += self.bias
        self.out = out
        return out

    def get_h_value(self, instance):
        return self.function(self.get_net_value(instance))

    def fit(self, inputs, target):
        if self.update_counter >= self.epoch:
            return 1
        else:
            output = 0
            for instance in inputs:
                output += self.get_net_value(instance)
            if self.get_h_value(output) - target == 0:
                return 1
            else:
                return 0
```

شکل۳: پیاده سازی perseptron در پایتون

برای حل این مسئله ۰.۲ =learning rate و 100 = epoch در نظر گرفتیم. و کلاس بندی را انجام دادیم.