به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

شبکه های عصبی و یادگیری عمیق

تمرین سری 1

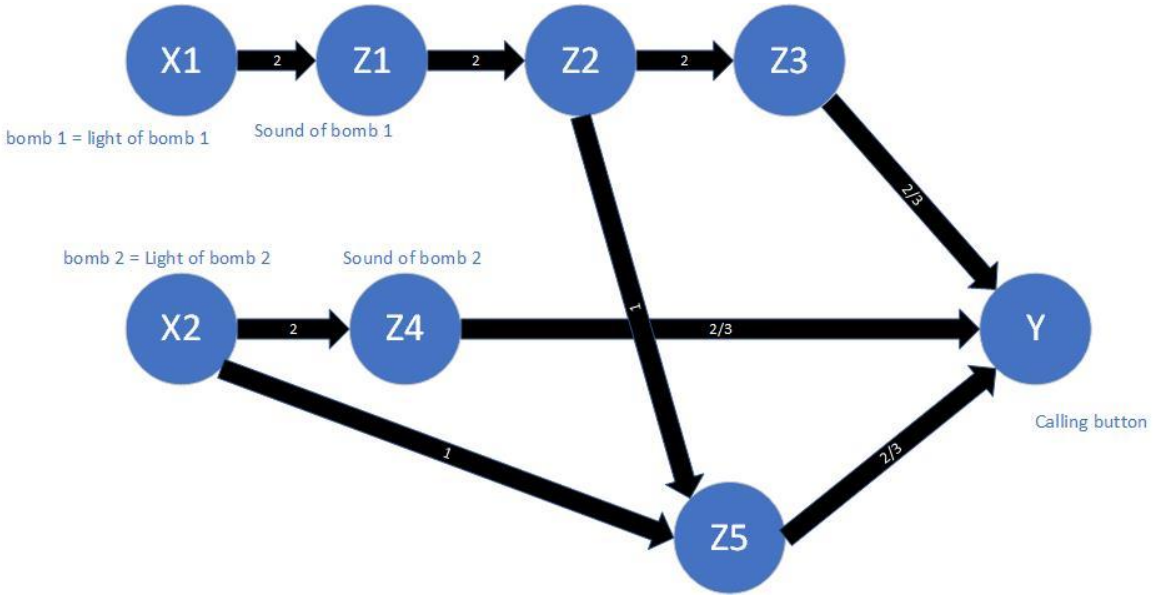
نام و نام خانوادگی	امید واهب
شماره دانشجویی	۸۱۰۱۹۶۵۸۲
تاریخ ارسال گزارش	۱۴۰۰/۰۱/۱۲

فهرست گزارش سوالات

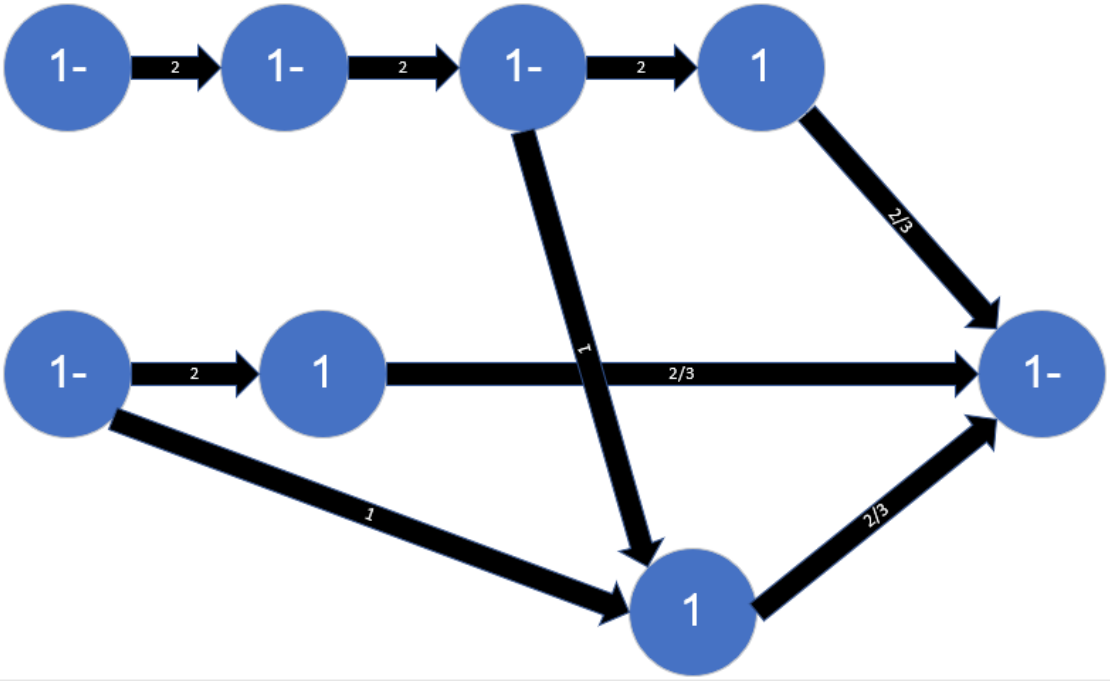
سوال ۱ – McCulloch-Pitts

در این سوال قصد داریم تا با طراحی شبکه ای از نورون های ساده McCulloch-Pitts سیستمی معادل فردی را طراحی کنیم که در صورتی که انفجار ۲ بمب با فاصله ی زمانی ۲ step زمانی را تشخیص داد دکمه ای را بزند. برای حل این سوال از مثال سورس استفاده می کنیم با این تفاوت که سیگنال ها را به جای اینکه باینری فرض کنیم 1 و 1- می گیریم(باینری باشد هم مشکلی ندارد صرفا برای فهم بهتر فرض شده). Threshold نیز برابر ۲ است و رفتار نورون مانند قبل است.

الف و ج) شبکه زیر را طراحی کردیم. همانطور که مشخص است انفجار بمب اول با دیدن نور حاصل از انفجار آن یکی است و توسط نورون X1 تشخیص داده می شود. نورون X2 نیز همین عملکرد را برای بمب دوم دارد. حال به بررسی عملکرد این شبکه می پردازیم. ابتدا خروجی همه ی نورون ها 1- است و در زمان صفر X1 یک می شود و به تبع آن در تایم استپ اول Z1 یک می شود و در تایم استپ دوم Z2 یک می شود البته X1 و Z1 در این زمان برابر 1- شده اند. همچنین می دانیم که در همین زمن بمب دوم نیز منفجر می شود و در نتیجه ی نور آن X2 یک می شود.



در تایم استپ سوم همانطور که در شکل زیر نیز قابل مشاهده است Z3 و Z4 و Z4 همگی برابر یک شده اند. Z3 به این دلیل شده که در زمان قبلی Z2 یک بوده. دلیل یک شدن Z4 در این تایم استپ نیز یک بودن X2 در تایم استپ دوم است. Z5 نیز که نورونی است که فاصله ی زمانی ۲ استپ بین دو انفجار را نشان می دهد، به دلیل یک بودن X2 و Z2 در تایم استپ قبلی یک شده است پس هر سه یال وارد بر Y در تایم استپ سوم برابر یک شدند پس در تایم استپ چهارم Y برابر یک می شود به عبارت دیگر اپراتور ما 2 استپ زمانی بعد از انفجار دوم و 4 استپ زمانی بعد از انفجار اول دکمه را می تواند فشار دهد. همچنین واضح است که در هیچ حالت دیگری Y برابر یک نمی شود زیرا باید فاصله ی ۲ انفجار ۲ استپ زمانی باشد تا Z5 برابر یک شود و Y تنها وقتی یک می شود که Z5 یک می شود.



ب)

Y(t-4) = Z3(t-3) and Z4(t-3) and Z5(t-3)

Z3(t-3) = Z2(t-2) ; Z2(t-2) = Z1(t-1) ; Z1(t-1) = X(t)

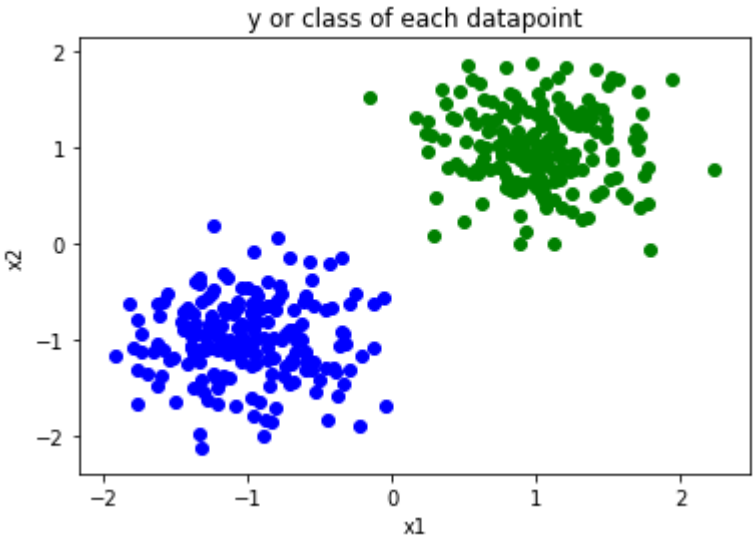
Z4(t-3) = X2(t-2)

Z5(t-3) = X2(t-2) and Z2(t-2); Z2(t-2) = Z1(t-1) ; Z1(t-1) = X(t)

سوال ۲ – Perceptron

در این سوال قصد داریم به کمک یک نورون و قاعده ی پرسپترون داده هایی را classify کنیم. هر سطر دیتاست ۳ ستون دارد که ۲تای اول ویژگی و آخری کلاس را نشان می دهد. کد و توضیحات کد این سوال در فایل Q2-Perceptron.ipynb قرار دارد.

الف) پس از خواندن داده ها از فایل داده شده اقدام به رسم scatterplot آنها می کنیم.



شکل ۱ – scatterplot داده ها

ب) حال داده را به train و test تقسیم می کنیم و نورون را تعریف می کنیم. تابع activation را تعریف می کنیم که با توجه به threshold و مقداری که می گیرد مثل تابع sign مقادیر 0 یا 1- یا 1 را بر می گرداند.

$$\text{net} = w_1x_1 + \cdots . + w_ix_i + \cdots + w_nx_n + b$$

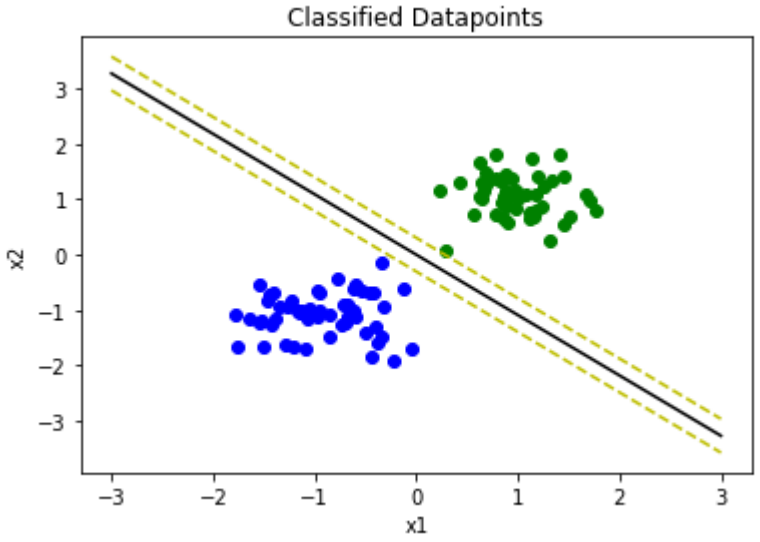
$$(\#) \quad h = \begin{cases} 1 & \text{if} \quad \text{net} > \theta \quad \text{Active} \\ 0 & \text{if} \quad |\,\text{net}\,| < \theta \quad \text{non - descision} \\ -1 & \text{if} \quad \text{net} < -\theta \quad \text{Passive} \end{cases}$$

$$\theta : \text{non} - \text{negative threshold}$$

سپس مقادیر اولیه وزن و بایاس را صفر قرار داده و نرخ یادگیری را ۰.۲ می گیریم. مقادیر دیگری نیز میتوانستیم قرار دهیم ولی سعی شده مقداری انتخاب شود که نه خیلی بزرگ باشد که واگرا شود نه خیلی کوچک که نیاز به تعداد زیادی epoch باشد. سپس حلقه ی اصلی را می نویسیم و ۱۰ ایپاک train می کنیم و وزن ها را آپدیت می کنیم به کمک update rule زیر:

$$b(new) = b(old) + \alpha \cdot t \qquad w_i(new) = w_i(old) + \alpha x_i t$$

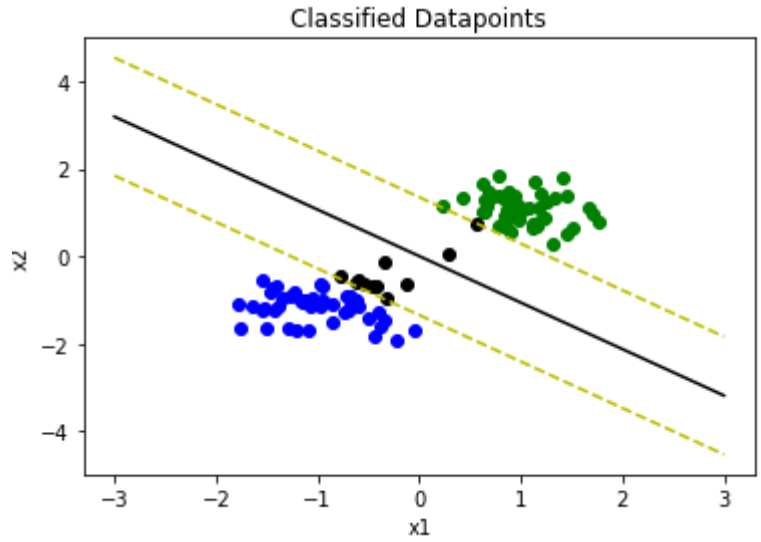
ج) حال به کمک یک تابع predict و یک حلقه تعداد prediction های درست نورون برای داده های تست را حساب می کنیم. به دقت ۱۰۰ درصد رسیدیم از scatterplot هم مشخص بود که این دقت دست یافتنی است زیرا داده ها به سادگی با یک خط جداپذیرند.



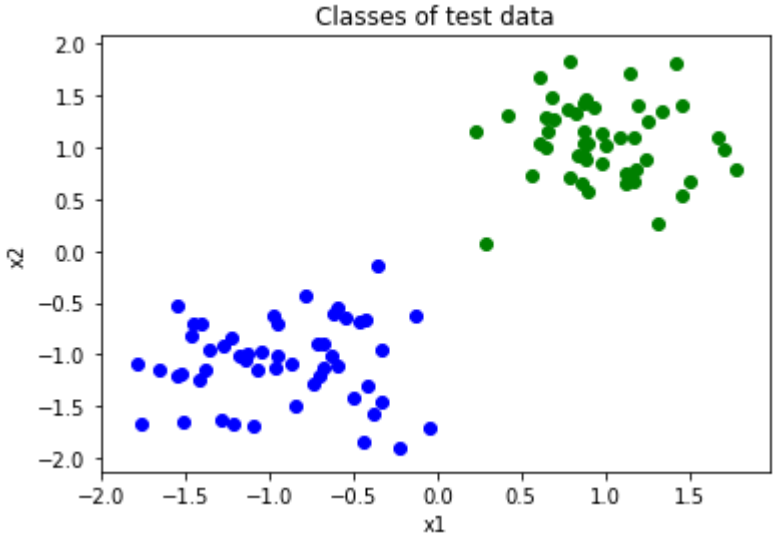
شکل ۲ – scatterplot داده ها و خط های جداکننده برای theta=0.2

همچنین خطوط جدا کننده را نیز رسم می کنیم و می بینیم که به خوبی threshold انتخابی ما دو کلاس را از هم جدا کرده است.

د) حال با theta برابر ۲۰۰ دو قسمت قبل را انجام می دهیم و میبینیم که دقت کاهش یافته زیرا تعدادی از داده ها به دلیل زیاد شدن threshold و فاصله گرفتن مرزها از خط وسطی درست تشخیص داده نشده اند. این اتفاق در شکل های زیر نیز قابل تشخیص است:



شکل 3 – scatterplot داده ها و خط های جداکننده برای theta=200



شکل 4 – scatterplot داده های تست و رنگ آمیزی درست آنها طبق کلاسشان

در شکل ۳ داده های مشکی تشخیص داده نشده اند. رنگ آمیزی درست این داده ها در شکل ۴ قابل مشاهده است. اگر اندازه Threshold بزرگ باشد این به معنای تغییر بزرگی در بایاس میباشد که در صورت آپدیت کردن و رسیدن به وزن های درست را سختتر خواهد کرد به epoch های بسیار بیشتری نیاز خواهیم داشت. همچنین ممکن است با threshold های بزرگ اصلا امکان جداسازی وجود نداشته باشد.

سوال ۳ – Adaline

در این سوال قصد داریم به کمک یک نورون adaline داده هایی را که خودمان به کمک واریانس و میانگین های داده شده و توزیع نرمال تولید کرده ایم، classify کنیم. هر سطر دیتاست ۳ ستون دارد که ۲تای اول ویژگی و آخری کلاس را نشان می دهد. کد و توضیحات کد این سوال در فایل Q3-Adaline.ipynb قرار دارد.

الف) شباهت آنها این است که هر دو مرز خطی دارند که دو کلاس را از هم جدا می کند. تفاوت اساسی آنها هم این است که adaline برای آپدیت کردن بجای target از predicted استفاده می کند و هدف مستقیم آن یادگیری در راستای کم کردن خطا پیشبینی target ها نیست. تفاوت دیگر این است که adaline معمولی(بدون استفاده از الگوریتم های خاص) تضمین همگرایی ندارد ولی perceptron دارد. علت هم تفاوت آنها در چیزی است که با آن وزن ها را آپدیت می کنند، است.

ب) ابتدا داده ها را generate می کنیم که توضیح آنها در فایل ژوپیتر هست. نتیجه هم در scatterplot های زیر است که دو کلاس با دو رنگ مختلف نشان داده شده اند. نتیجه مطابق انتظار و مقادیر فرض شده است:

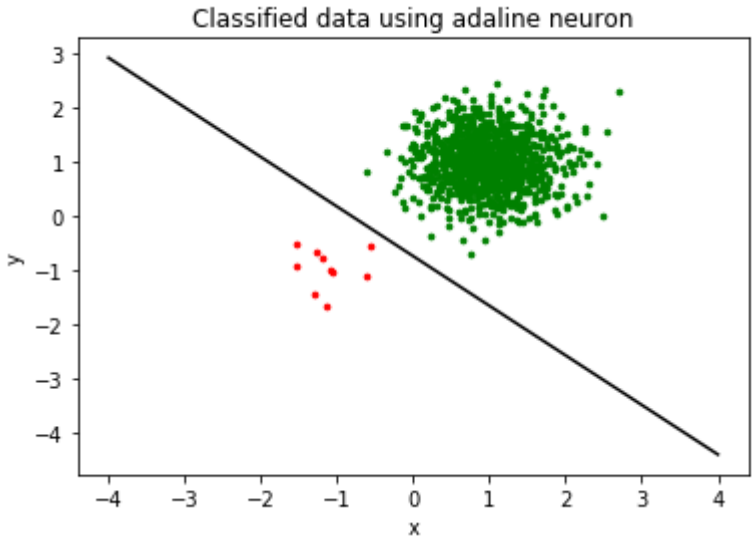
شکل ۵ – داده های generate شده برای مثال اول

شکل ۶ – داده های generate شده برای مثال دوم

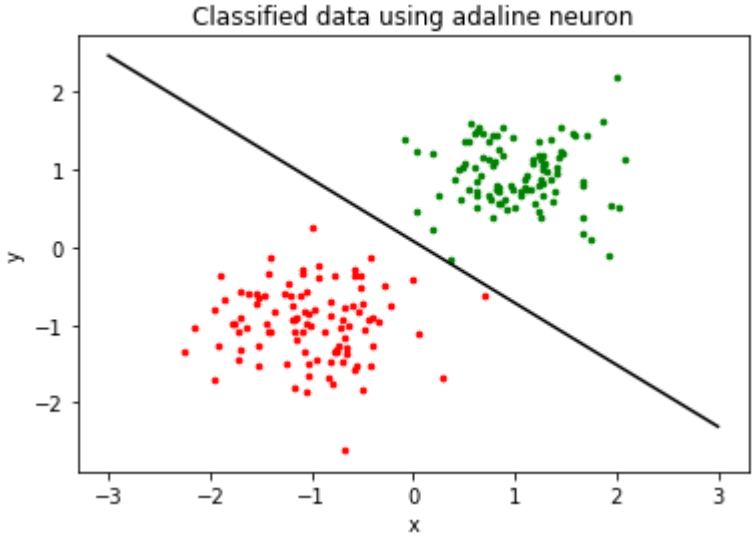
در این الگوریتم net را به صورت $\sum_{i=1}^{n} w_i x_i + b$ بنویسیم و با تشکیل تابع activation اگر مثبت بود ۱ و در غیر این صورت 1- را برگردانیم. حال به کمک قانون آپدیت زیر وزن ها و بایاس را به روز می کنیم:

$$w_i(new) = w_i(old) + \alpha(t_i - net)x_i$$
$$b_i(new) = b_i(old) + \alpha(t_i - net)$$

در فایل ژوپیتر موجود نیز همین کارها را کردیم و تعریف ها را کردیم. نکته ی قابل توجه مقادیر اولیه وزن ها و بایاس بود که رندوم انجام شد. همچنین مقدار نرخ یادگیری هم سعی شد طوری انتخاب شود که نه خیلی کند باشد نه زیادی بزرگ که منجر به واگرایی شود. نتایج حاصل از این کد و خطوط جدا کننده در زیر قابل مشاهده است:



شکل ۷ – خط جداکننده مثال اول و داده ها



شکل ۸ – خط جداکننده مثال دوم و داده ها

می توان مشاهده کرد که به خوبی عمل classification انجام شده است البته تضمین همگرایی نداشتیم که جلوتر بیشتر توضیح داده خواهد شد. همچنین می توان دید که مثل perceptron نمی توانیم با تغییر threshold فاصله از داده ها دستی عوض کنیم.

ج) خیر در روش دلتا تضمین همگرایی نداریم زیرا از تفاضل net و t به جای t و h استفاده شده است. راه حل این مشکل استفاده از تابع soft sign یا توابع مشابه است به جای sign

د) در این روش تابع هزینه ای مثل زیر را بهینه می کنیم:

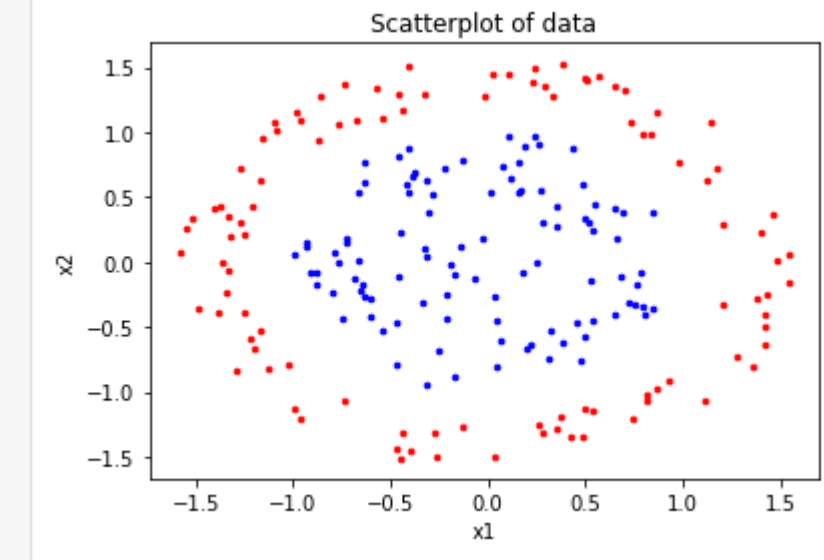
$$J_p(w,b) = 0.5 * \left(t_p - net\left(x_p, w, b\right)\right)^2$$

که با نوشتن روابط ریاضی می توانیم تغییرات وزن ها را برابر $\alpha(t-net)x_i$ و تغییر بایاس را برابر $\alpha(t-net)$ بنویسیم که به وضوح تاثیر نرخ یادگیری یا همان آلفا در آن مشخص است. اگر مقدار بزرگی انتخاب شود منجر به واگرایی می شود و اگر کم باشد نیاز به تعداد epoch زیاد خواهد بود.

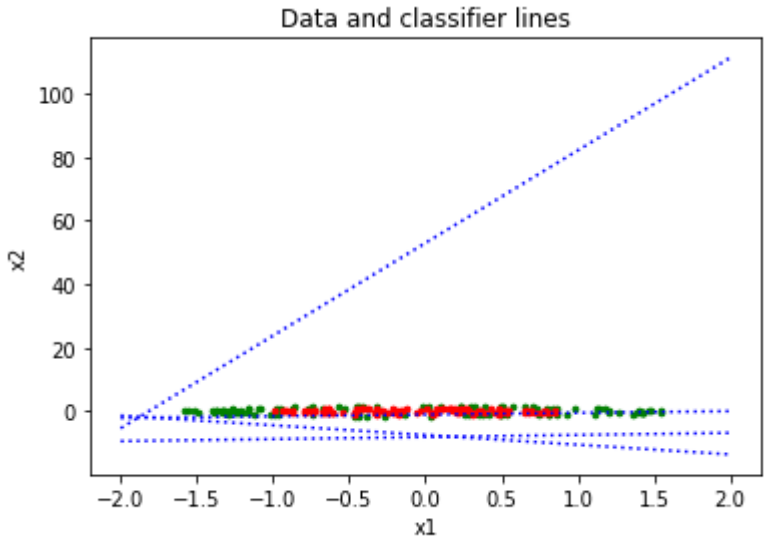
سوال ۴ – Madaline

در این سوال قصد داریم به کمک یک نورون Madaline داده هایی را که در یک فایل csv ذخیره شده اند را classify کنیم. هر سطر دیتاست ۳ ستون دارد که ۲تای اول ویژگی و آخری کلاس را نشان می دهد. کد و توضیحات کد این سوال در فایل Q4-Madaline.ipynb قرار دارد.

الف و ب و ج) ابتدا داده ها را از فایل می خوانیم. برای کار راحتتر توسط کتابخانه pandas در فایل csv تغییری می دهیم و label برای ستون ها تعریف می کنیم.



شکل ۹ – داده های موجود



شکل ۱۰ - خطوط جداکننده

متاسفانه به دلیل ماهیت تصادفی این الگوریتم موفق به گرفتن نتیجه نشدم و خطوط خیلی پرت اند.