نام و نام خانوادگی	علی رنجبری – امیرحسین علیزاد
شماره دانشجویی	۸۱۰۱۹۸۵۷۰ - ۸۱۰۱۹۷۵۴۶
تاریخ ارسال گزارش	۱۴۰۱/۰۸/۱۰

به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق
تمرین اول

فهرست

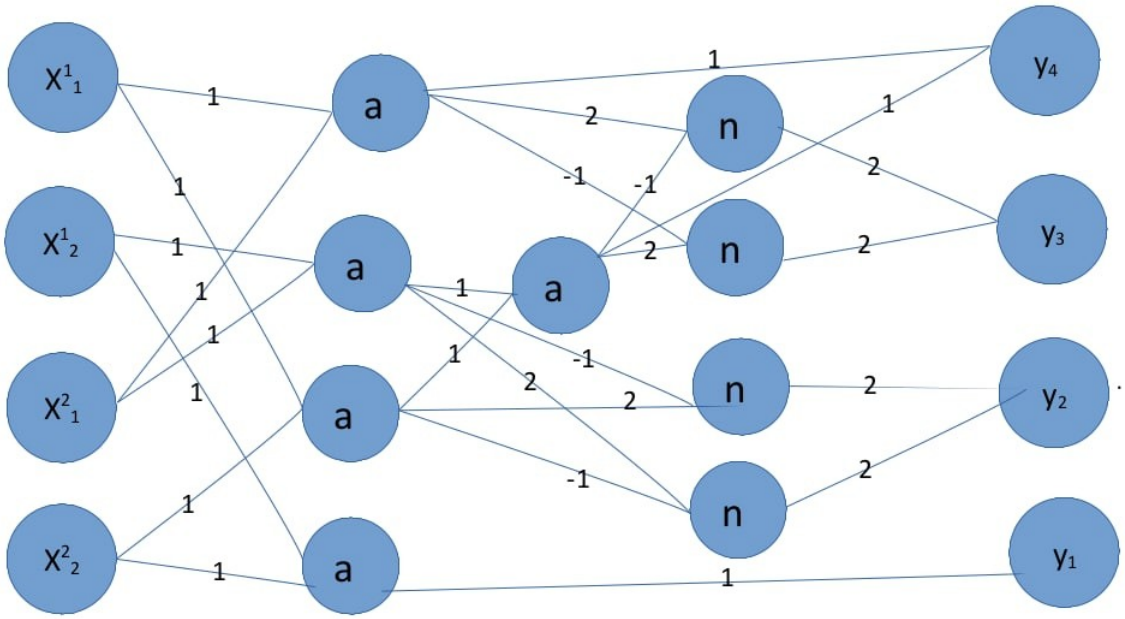
فهرست 1
پاسخ ۱. شبکه عصبی Mcculloch-Pitts 1
۱-۱.ضرب کننده باینری دو بیتی 1
پاسخ ۲ – AdaLine and MadaLine 3
۲-۱. **AdaLine** 3
داده‌ها به صورتی که سؤال خواسته بود تعریف شدند و به صورت زیر شدند: 3
شکل ۳. نمودار پراکندگی دو دسته داده (میانگین ۱ و -۱ و انجراف معیار ۰.۳) 3
شکل ۴و۵ . نمودار تغیرات خطا در تابع adaline و خط جدا کننده دو تابع بعد از train 3
شکل ۶ . نمودار پراکندگی دو دسته داده (میانگین ۰ و ۲ و انجراف معیار ۰.۶ و ۰.۸) 4
شکل ۷ . خط جدا کننده دو تابع بعد از train کردن توسط adaline 4
۲-۲. **MadaLine** 5
شکل ۸ . نمودار پراکندگی دو دسته داده برای شبکه MadaLine 5
شکل ۹ . شبکه نمونه MadaLine 5
شکل ۱۰ . خطهای تولید شده سه شبکه متفاوت (از راست به چپ ۲، ۴ و ۸ نورون) 6
شکل ۱۱ . تعداد epoch ها و دقت شبکه خروجه (بالا راست ۲ بالا چپ ۴ و پایین راست ۸ نورون) 6
پاسخ ۴ – MLP 7
۴-۱. **Multi Layer Perceptron** 7
شکل ۱۲ . نمودار correlation بین مقادیر موجود در داده ها 7
شکل ۱۳ . نمودار پراکندگی قیمت خانه ها 8
شکل ۱۴. نمودار قیمت خانه ها و متراژ آن ها 8

پاسخ 1. شبکه عصبی Mcculloch-Pitts

۱-۱.ضرب کننده باینری دو بیتی

الف) شبکه خروجی به این صورت رسم شد:

Theta = 2



شکل ۱. شبکه عصبی ضرب کننده باینری دو بیتی

بعد از بررسی نوشتن عبارت هر خروجی شبکه نهایی مشابه شکل بالا شد. همانطور که نوشته شد تتا در همه نورون ها ۲ است. همچنین نورون هایی که با a مشخص شده‌اند عمل‌کرد and را دارند و نورون هایی که با n نمایش داده شده‌اند به همراه نورون کنارشان و نورون خروجی بعدیشان عمل‌کرد and not را دارند. ساختار شبکه هم یک ساختار منطقی است مثلاً برای کم ارزش ترین بیت خروجی (y_1) فقط کافی است دو بیت X^1_2 و X^2_2 همزمان یک باشد تا آن هم یک باشد در غیر این صورت صفر است. بقیه هم به همین صورت فقط کمی پیچیده‌تر است اما کلیت هر بیت خروجی به همین صورت تعیین می‌شود.

ب) ابتدا یک تابع mcculloch-pitts رسم شد که وزن ها و تتا را میگیرد و یک تابع خروجی میدهد. سپس با توجه به شکل ۱ شبکه رسم شد و بعد روی تمام ورودی های ممکن خروجی گرفته شد که نتیجه همه در زیر آمده:

```
[0, 0] * [0, 0] = [0, 0, 0, 0] ⟶ 0 * 0 = 0
[0, 0] * [0, 1] = [0, 0, 0, 0] ⟶ 0 * 1 = 0
[0, 0] * [1, 0] = [0, 0, 0, 0] ⟶ 0 * 2 = 0
[0, 0] * [1, 1] = [0, 0, 0, 0] ⟶ 0 * 3 = 0
[0, 1] * [0, 0] = [0, 0, 0, 0] ⟶ 1 * 0 = 0
[0, 1] * [0, 1] = [0, 0, 0, 1] ⟶ 1 * 1 = 1
[0, 1] * [1, 0] = [0, 0, 1, 0] ⟶ 1 * 2 = 2
[0, 1] * [1, 1] = [0, 0, 1, 1] ⟶ 1 * 3 = 3
[1, 0] * [0, 0] = [0, 0, 0, 0] ⟶ 2 * 0 = 0
[1, 0] * [0, 1] = [0, 0, 1, 0] ⟶ 2 * 1 = 2
[1, 0] * [1, 0] = [0, 1, 0, 0] ⟶ 2 * 2 = 4
[1, 0] * [1, 1] = [0, 1, 1, 0] ⟶ 2 * 3 = 6
[1, 1] * [0, 0] = [0, 0, 0, 0] ⟶ 3 * 0 = 0
[1, 1] * [0, 1] = [0, 0, 1, 1] ⟶ 3 * 1 = 3
[1, 1] * [1, 0] = [0, 1, 1, 0] ⟶ 3 * 2 = 6
[1, 1] * [1, 1] = [1, 0, 0, 1] ⟶ 3 * 3 = 9
```

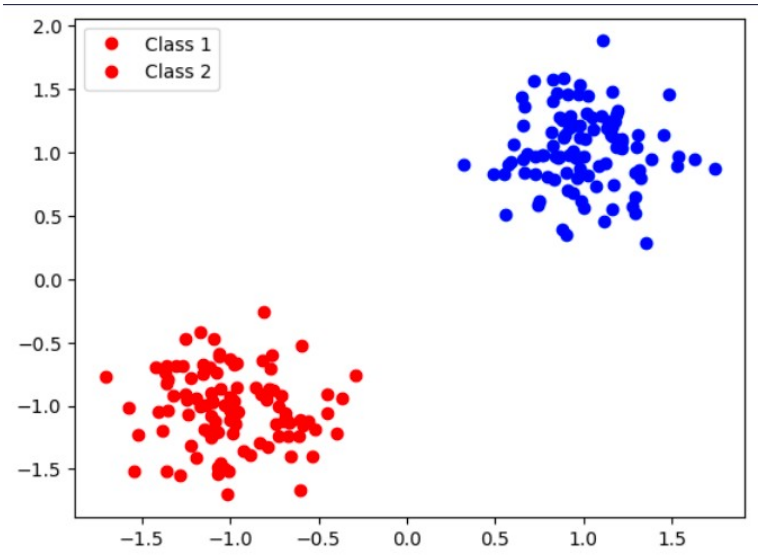
شکل ۲. خـروجی تـابع multiplier روی تمام ورودی های ممکن

همانطور که در شکل ۲ هم مشخص است همه خروجی ها صحیح هستند.

پاسخ ۲ – AdaLine and MadaLine

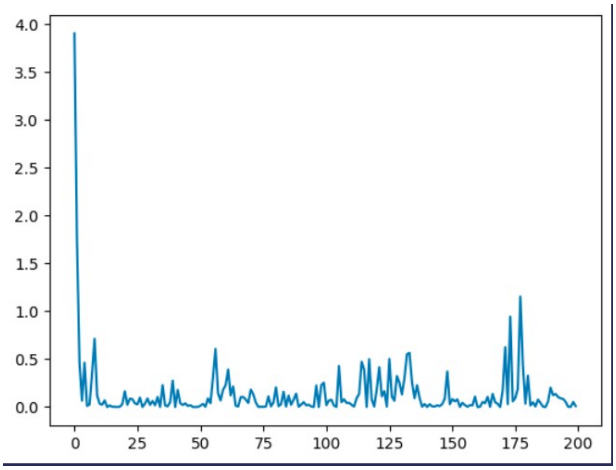
۲-۱. AdaLine

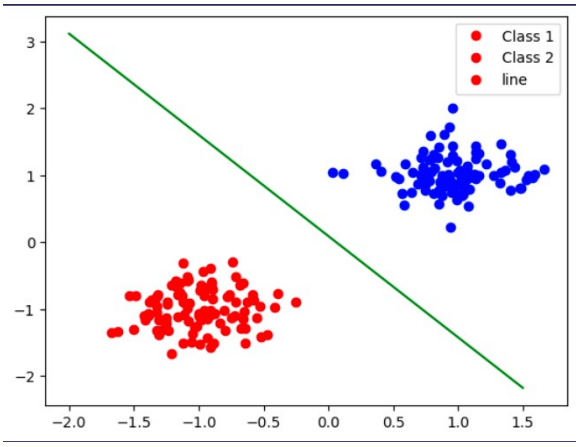
داده‌ها به صورتی که سؤال خواسته بود تعریف شدند و به صورت زیر شدند:



شکل ۳. نمـودار پراکنـدگی
دو دسته داده (میانگین ۱ و -۱ و انحراف معیار ۰.۳)

سپس با استفاده از تابع Adaline که نوشته شده بود train شدند و نمودار تابع خطا به صورت زیر شد:

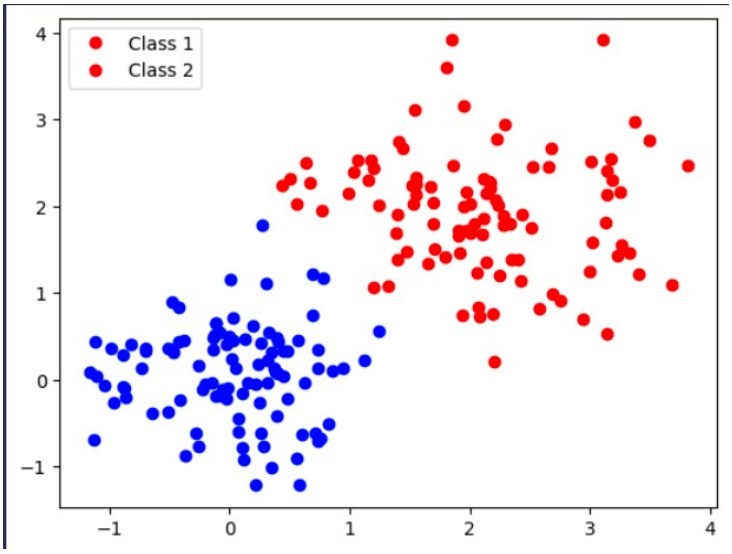




شکل ۴و۵ . نمودار تغیرات خطا در تابع adaline و خط جدا کننده دو تابع بعد از train

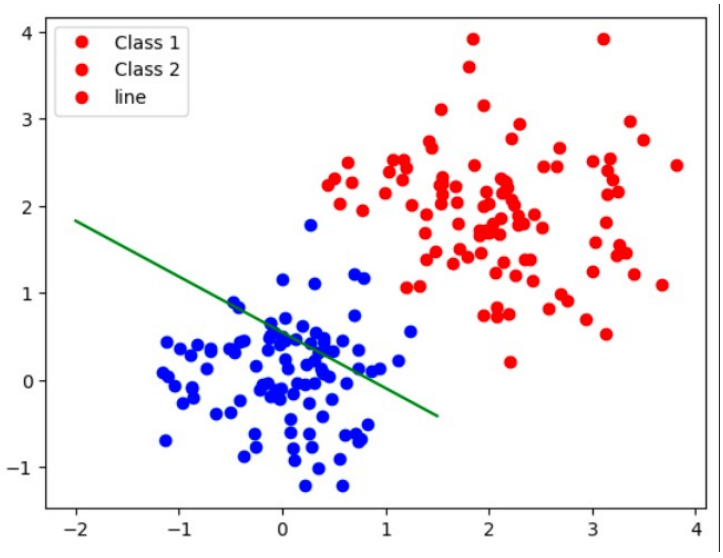
در شکل ۵ مشخص است که adaline توانسته به خوبه داده‌ها را با دقت صد درصد جدا کند.

اکنون دو دسته داده دیگر را تعریف میکنیم که بیشتر به هم نزدیک هستند جداسازی سخت تری دارند



شکل ۶. نمودار پراکندگی دو دسته داده (میانگین ۰ و ۲ و انحراف معیار ۰.۶ و ۰.۸)

بعد از یک دور دیدن داده‌ها (یک epoch)ـ adaline توانست با دقت ۶۰ درصد بصورت زیر داده‌ها را جـدا کند.

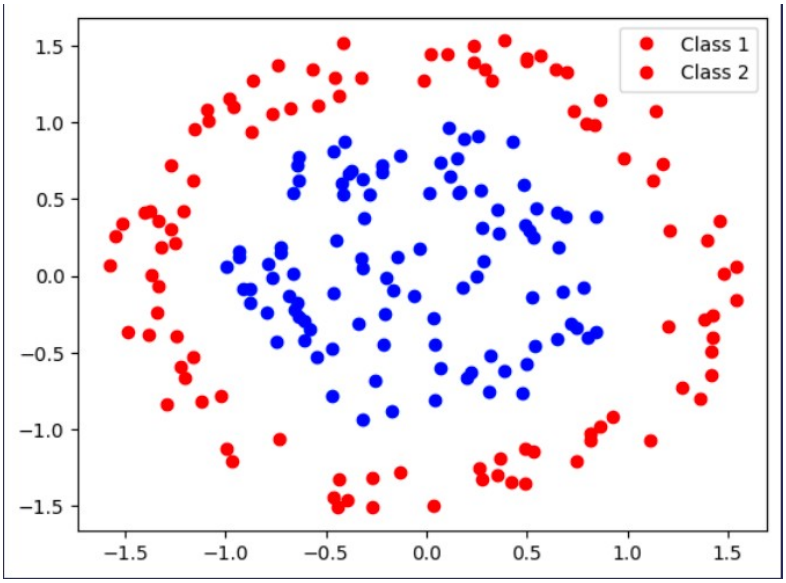


شکل ۷. خط جدا کننده دو تابع بعد از train کردن توسط adaline

همانطور که در شکل و نتایج هم واضح بود وقتی انحراف معیار داده‌ها بیشتر شد و داده‌ها دیگر به راحتی با یک خط جدا پذیر نبود adaline دیگر نتوانست دو دسته داده را به خوبی از هم جدا کند. که انتظار همین نتیجه هم از الگوریتم adaline میرفت.

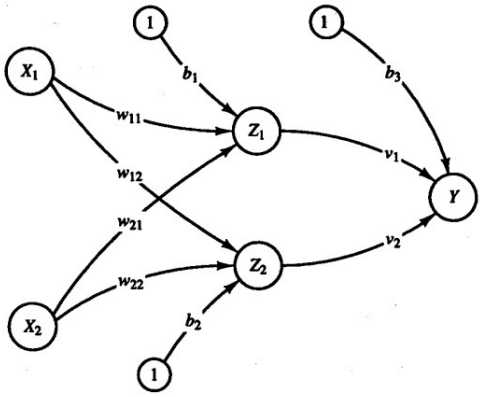
MadaLine .۲-۲

در این بخش پس از خواندن داده‌ها که دیگر بصورت خطی نمیتوان دسته بندی کرد (شکل ۷) میخواهیم به کمک شبکه MadaLine و الگوریتم MRI آن را ترین کنیم.



شکل ۸ . نمودار پراکندگی دو دسته داده برای شبکه MadaLine

شبکه MadaLine مانند شبکه AdaLine است با این تفاوت که یک لایه مخفی هم دارد(شکل ۸):

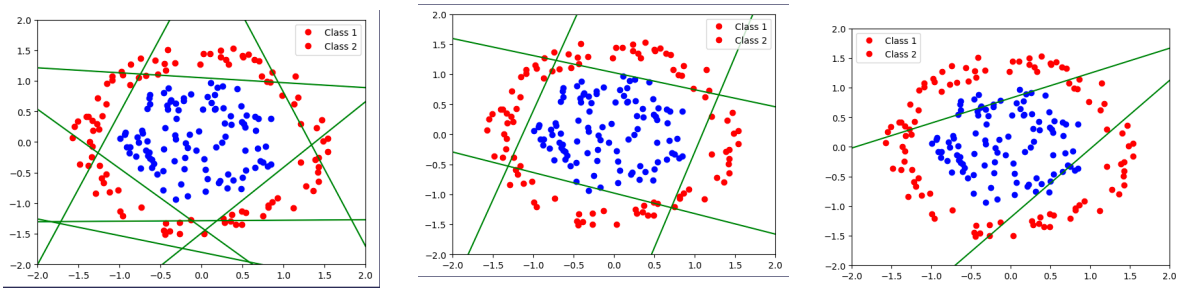


شکل ۹ . شبکه نمونه MadaLine

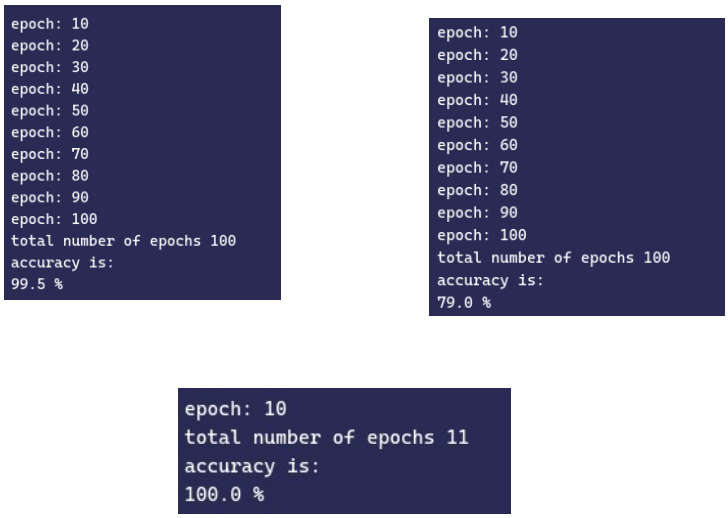
ما در اینجا این شبکه را با الگوریتم MRI آموزش دادیم این الگوریتم به این صورت است:

وزن های قسمت دوم شبکه (..,v1, v2) طوری تعیین می‌شوند که به صورت And عمل کنند یعنی اگر همه Z_i ها ۱ باشند آنگاه Y هم ۱ است در غیر این صورت -۱ است. بنابراین ما در شبکه فقط W_{ij} ها را آموزش میدهیم. که جزئیات بیشتری دارد که در اینجا فقط یک نگاه اجمالی انداختیم.

سپس سراغ آموزش شبکه می‌رویم در اینجا از سه شبکه استفاده شد که به ترتیب ۲، ۴ و ۸ نورون در لایه پنهان داشتند:



شکل ۱۰ . خط‌های تولید شده سه شبکه متفاوت (از راست به چپ ۲، ۴ و ۸ نورون)



شکل ۱۱ . تعداد epoch ها و دقت شبکه خروجه (بالا راست ۲ بالا چپ ۴ و پایین راست ۸ نورون)

همانطور که در شکل‌ها نتایج مشخص است هر چه تعداد نورون ها افزایش پیدا میکند دقت شبکه بیشتر می‌شود هم چنین تعداد epoch ها برای رسیدن به دقت بالا هم کمتر می‌شود. مثلاً برای شبکه آخر با ۸ نورون توانسته با ۱۱ epoch به دقت ۱۰۰ درصد برصد که در شکل ۸ هم مشخص است که فقط ۵ خط برای به طور کامل تقسیم کردن این داده‌ها کافی بود و بقیه خط ها دیگر نیاز نشدند. یا در شبکه اول با اینکه توانسته به ۷۹ درصد دقت برسد اما واضح است که اگر به این شبکه بیشتر از ۱۰۰ epoch هم اجازه آموزش داده میشد بازهم به دقت ۱۰۰ درصد نمیرسید چون این نوع داده به هیچ وجه با دو خط قابل جدا سازی نیستند.

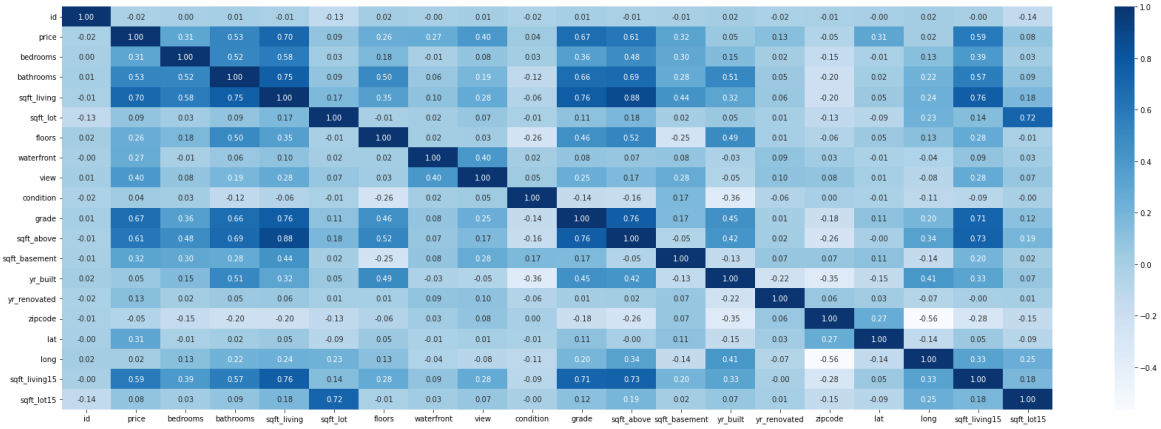
پاسخ ۴ – MLP

۴-۱. Multi Layer Perceptron

برای این بخش ابتدا با استفاده از تابع pandas.info() تعداد سطرها و ستون ها، نوع داده های موجود در ستون ها، نام ستون ها و تعداد مقادیر خالی در هر ستون را نشان می دهیم.

سپس با استفاده از توابع pandas.isna().sum() و pandas.isnull().sum() مجموع تعداد سطر هایی در هر ستون که مقدار NaN و یا Null دارند را مشخص می کنیم که این مقادیر برای تمامی ستون ها برابر صفر است و تمامی سطر ها و ستون ها شامل مقادیر قابل قبول هستند.

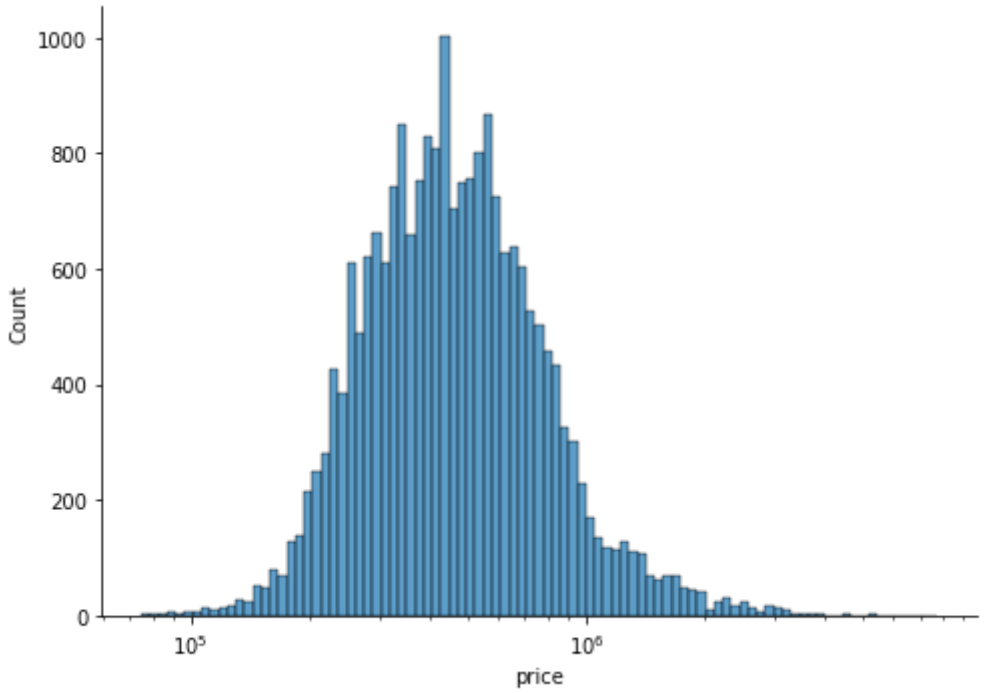
ابتدا correlation داده ها را با استفاده از تابع pandas.corr() محاسبه کرده و سپس با استفاده از کتابخانه seaborn و تابع heatmap این نمودار را نشان می دهیم.



شکل ۱۲ . نمودار correlation بین مقادیر موجود در داده ها

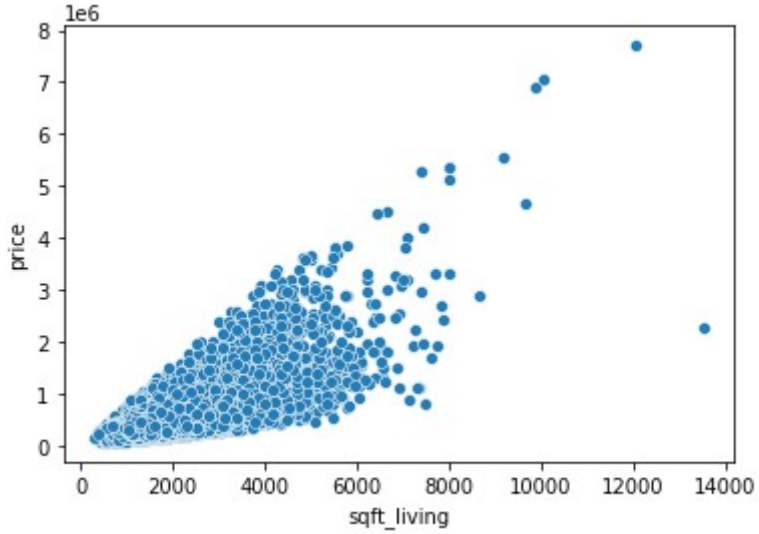
با توجه به این نمودار، مقدار sqrft_living با قیمت هر خانه بیشترین correlation را دارد که تا حدودی قابل حدس است که هر چه متراژ یک خانه بیشتر باشد، قیمت آن نیز بیشتر می شود و این فیچر بیشترین تاثیر را روی قیمت دارد.

سپس با استفاده از کتابخانه seaborn نمودار پراکندگی قیمت(در هر بازه قیمت چه تعداد خانه وجود دارد) و نمودار قیمت با فیچری که بیشترین correlation(متراژ-sqrft_living) را با آن دارد می کشیم.



شکل ۱۳ . نمودار پراکندگی قیمت خانه ها

این نمودار نشان می دهد که در هر بازه قیمت چه تعداد خانه وجود دارد.



شکل ۱۴. نمودار قیمت خانه ها و متراژ آن ها

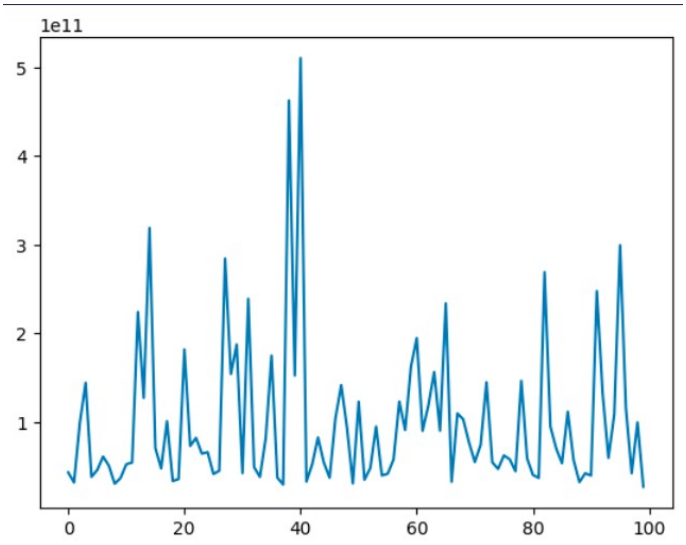
این نمودار نشان می دهد که به ازای هر مقدار sqrft_living آن خانه چه قیمتی دارد.

سپس با استفاده از تابع pandas.to_datetime() ستون date در داده ها را به فرم طبیعی تاریخ تبدیل می کنیم و سپس فیلد های year و month از این تاریخ را به داده های خود اضافه می کنیم.

با بررسی ستون های موجود در داده و همچنین نمودار correlation متوجه می شویم که پس از حذف ستون تاریخ، ستون های id و zipcode نیز تاثیری در پیش بینی قیمت خانه ندارند بنابراین این مقادیر را نیز از داده ها حذف می کنیم.

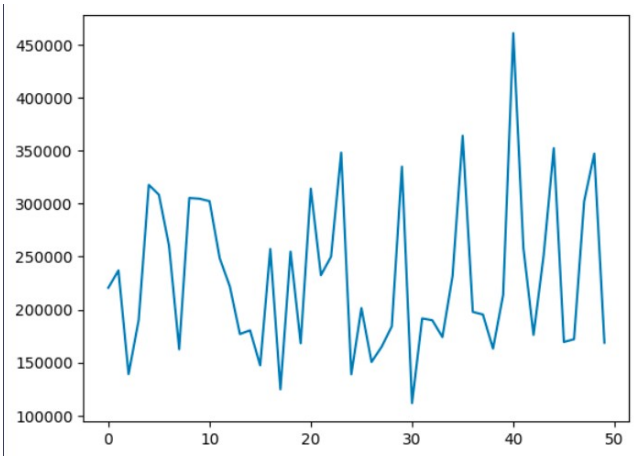
سپس داده های خود را به 2 دسته test و train تقسیم می کنیم که داده های train 80 درصد داده های ما را شامل می شود.

برای ترین ابتدا از MSELoss و SGD استفاده کردیم اما چون MSELoss خیلی بزرگ بود درست کار نمیکرد و Nan میداد. با کمی تغییر learning rate خطا را خروجی می‌داد اما خطا بسیار زیاد بود.



شکل ۱۵. تغییرات خطا شبکه عصبی اول

سپس از L1Loss و Adam استفاده کردیم و به میزان ۵۰ epoch ترین کردیم . و loss function مشابه زیر شد.



شکل ۱۶. تغییرات خطا شبکه عصبی دوم

خروجی شبکه برای دو ورودی نمونه :

```
    test_iter = iter(test_dataloader)
    x, y = next(test_iter)

    output = model2(x)
    output[0:2]
[176]  ✓ 0.5s
...  tensor([[462355.1041],
            [461214.0856]], grad_fn=<SliceBackward0>)

    y[0:2]
[177]  ✓ 0.5s
...  tensor([538000., 650000.])
```

شکل ۱۷. پیشبینی دو قیمت دو خانه ورودی نمونه (بالا پیشبینی و پایین قیمت واقعی)

میبینیم که قیمت های پیشبینی شده خیلی دور تر از قیمت واقعی نیستند و شبکه تا حدودی خوب کار میکند با اینکه باز هم خطای به نسبت بالایی دارد.