به نام خدا

امیررضا رجبی 983126

گزارش پیاده سازی شبکه عصبی

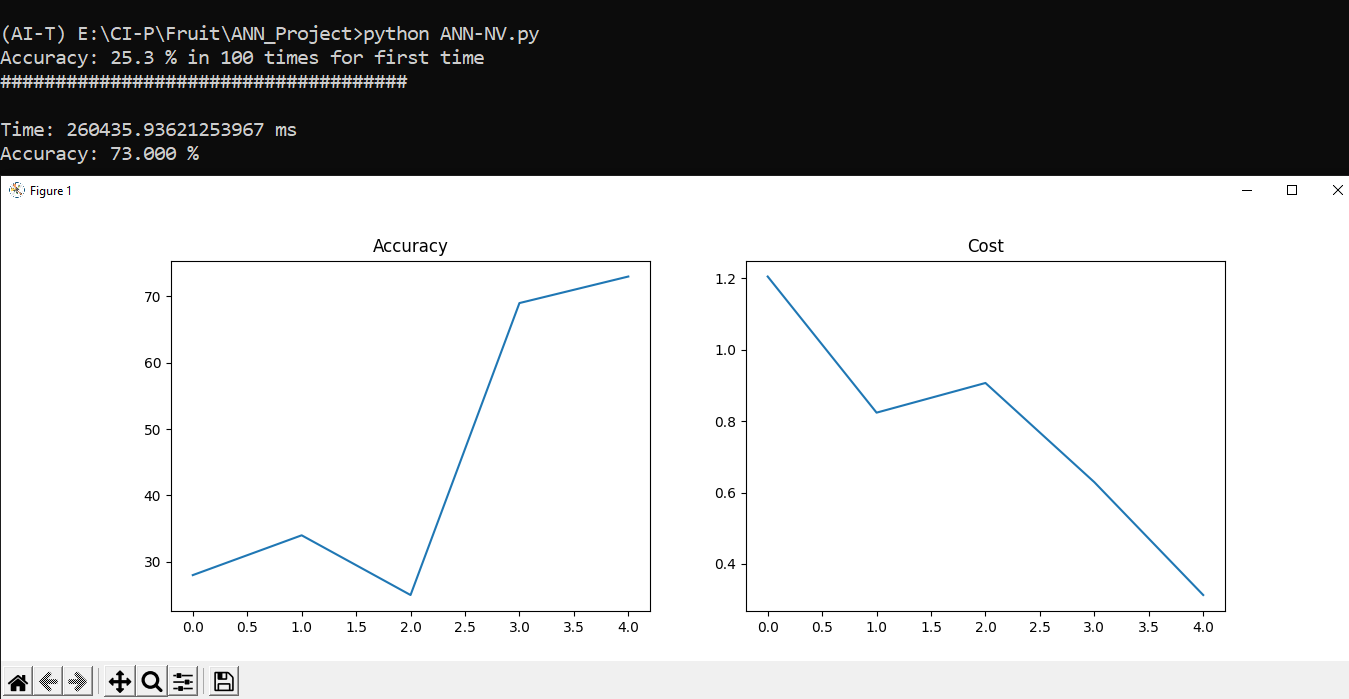
قدم دوم و سوم :

در قدم دوم قرار است شبکه ای که وزن دهی آن به صورت تصادفی است را آزمایش کرده و به طور میانگین باید دقت آن چیزی در حدود 25 درصد باشد

در قدم سوم ما قرار است شبکه خود را با 200 داده ای که درقدم دوم از آن استفاده کردیم آموزش دهیم اما با یک شرط آنکه بدون استفاده از وکتور کردن باشد

من کد این دو قسمت را یکی کرده و باهم اجرا میکنم.

ANN-NV.py



صد بار روی آن 200 داده با وزنای مختلف بدون لرن (قدم 2) پردیکت کرده که دقت میانگین آن 25.3 درصد درآمد

روند کاست نزولی است و روند دقت صعودی است و به 73 درصد دقت آن رسید

مدت زمان آن 260 ثانیه است

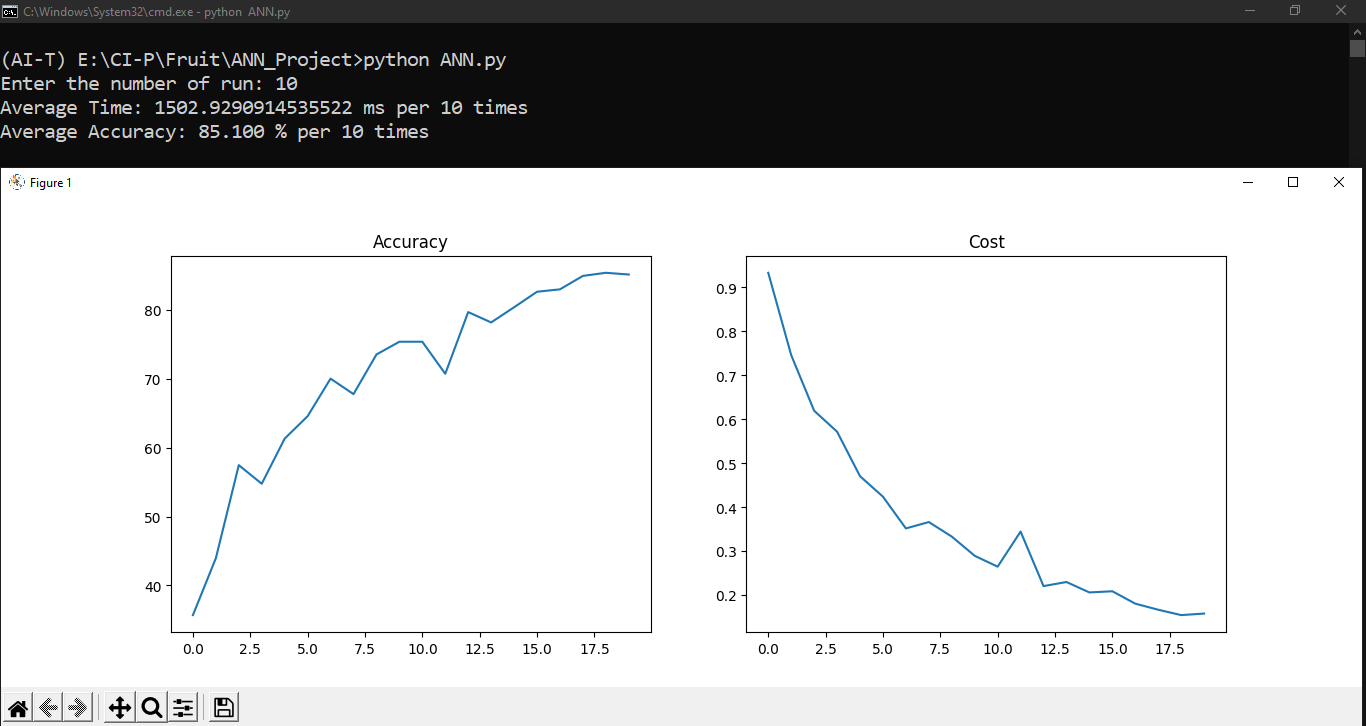
قدم چهارم :

ANN.py

در این مرحله ما باید با وکتورسازی زمان کد را بهبود بدهیم

پس کارهایی که باید انجام دهیم :

وکتور کردن و افزایش دادن ایپاک به 20 است ما کد رو 10 بر ران گرفته و میانگین رو میگیریم



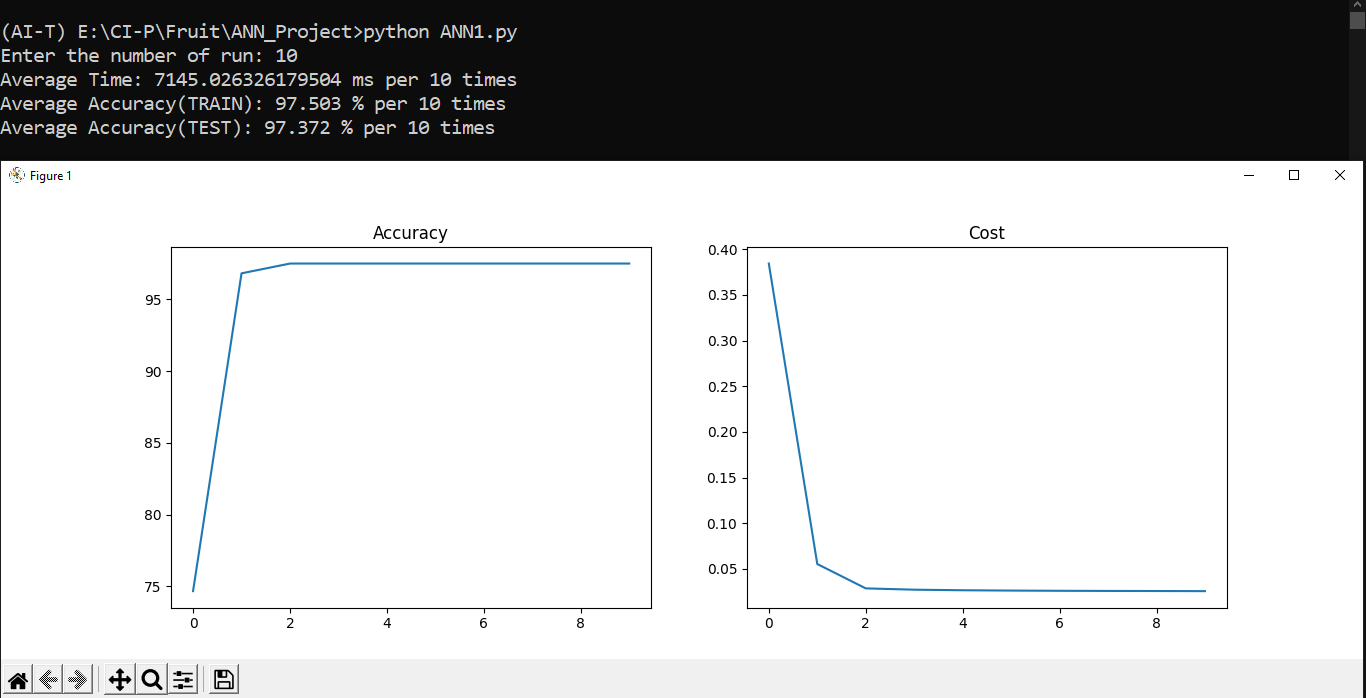
میانگین زمان اجرا به 1.5 ثانیه و دقت به 85.1 درصد بهبود یافت

از کاست و دقت برای هر ایپاک در هر ده ران میانگین گرفته وبعد پلات رو رسم کردم

قدم پنجم :

مراحل رو اجرا کرده و ده بار مانند قبل ران میگیریم

ANN1.py



به طور میانگین زمان اجرای آن 7 ثانیه شد

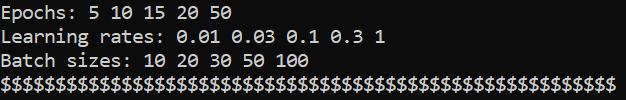
و به طور میانگین دقت مجموعه ترین آن 97.503 و در مجموعه تست 97.372 درصد شد

امتیازی اول :

من برای 125 حالت از ترکیب ایپاک و بچ سایز و لرنینگ ریت بدست آوردم

که 125 حالت تمامی تاپل های ممکن سه تایی از مقادیر پایین است که نتیجه آن در فایل زیر ذخیره شده است

Result\_extra1.txt , ANN\_Extra1.py



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 5 | 10 | 15 | 20 | 50 |
| Epoch | 6 | 9 | 8 | 11 | 17 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0.01 | 0.03 | 0.1 | 0.3 | 1 |
| L-R | 3 | 5 | 9 | 14 | 20 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 10 | 20 | 30 | 50 | 100 |
| B-Size | 17 | 12 | 10 | 7 | 5 |

در جدول بالا آنهایی که دقت تست بالای 98 را داشتند مقدارشان آورده شده

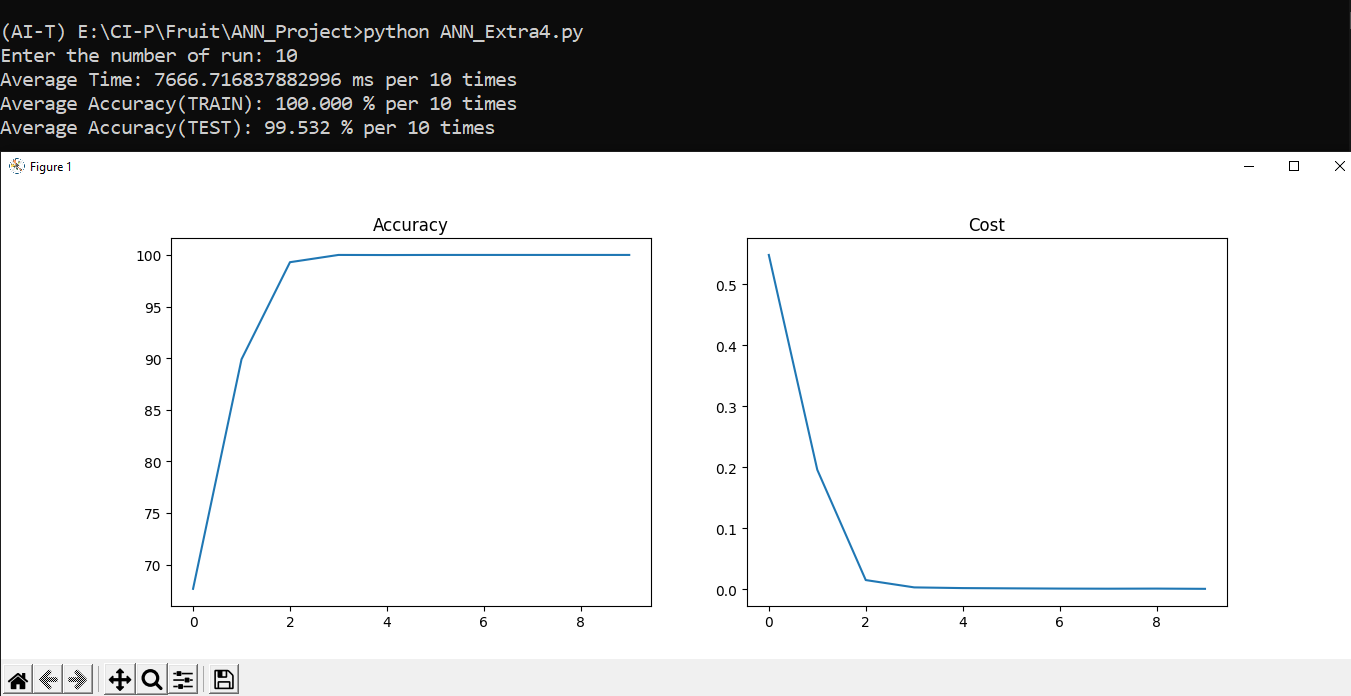
لرنینگ ریت بین 0.3 و 1 به سمت 1 بسیار مناسب است

برای ایپاک مقدار 50 مناسب است اما زمان را باید در نظر گرفت که در این پروژه چون خیلی تغییری نمیکند پس ماسب است اما باید اورفیت را در نظر گرفت

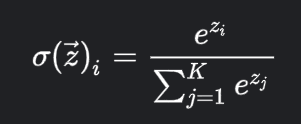
برای بچ سایز هم مقدار 10 مناسب مناسب است

امتیازی چهارم :

برای لایه آخر از تابع سافتمکس استفاده میکنیم و برای مقایسه مانند قدم پنجم عمل میکنیم

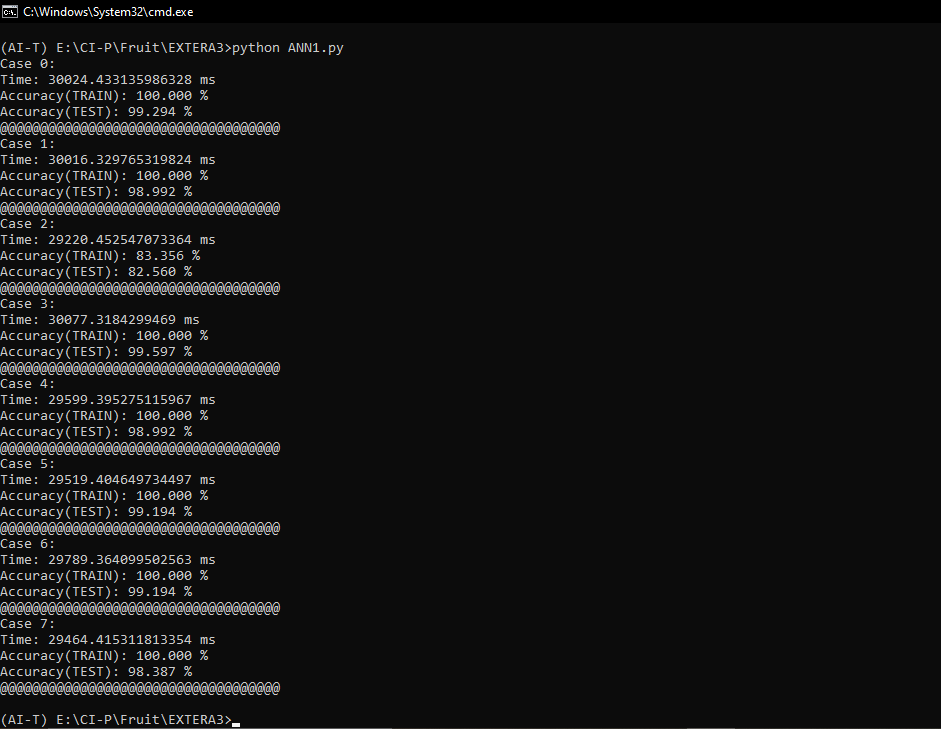


زمان یک مقدار بالاتر میرود چون تابع سافتمکس هزینه بیشتری دارد . میانگین ترین 100 درصد در ده ران و میانگین تست 99.532 است که جفتشون از قدم پنجم بهتر هستند . نتیجه که گرفتیم بهبود پیدا کرده به این علت که سافتمکس می آید اون اختلاف رو بیشتر میکند یعنی اونی که بیشترهست رو به 1 نزدیک و بقیه رو به صفر نزدیک میکند



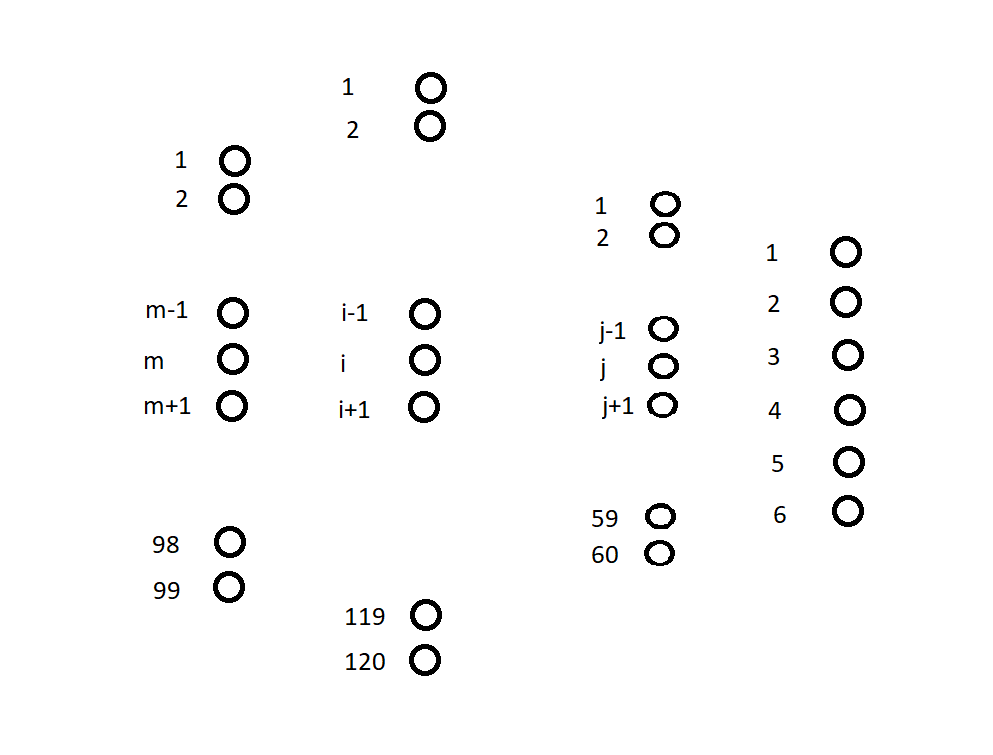
امتیازی سوم :

من دو میوه نارگیل و گیلاس از این دیتاست رو اضافه کردم



Learning\_rate = 1, batch\_size = 30, epoch = 30

مقادیر بالا و تعداد لایه ها و تعداد نورون ها در هر لایه با سعی و خطا بدست آمده اند



امتیازی دوم :

Momentum:

https://latex.codecogs.com/gif.latex?W_%7Bnew%7D%20%3D%20W_%7Bold%7D%20&plus;%20%5CDelta%20%7BW_%7Bnew%7D%7D

https://latex.codecogs.com/gif.latex?%5CDelta%20W_%7Bnew%7D%20%3D%20%5Calpha%20%5CDelta%20W_%7Bold%7D%20&plus;%20%5Ceta%20d%7BW_%7Bnew%7D%7D

در واقع داریم برای آپدیت این لایه از وزن لایه های از تغیرات لایه پیشین استفاده میکنیم و این روند رو تسریع میکند و جلوگیری میکند از توقف در مینیمم محلی که آلفا مقدار ممنتوم اس که عددی بین 0 و 1 است

