بسمه تعالی

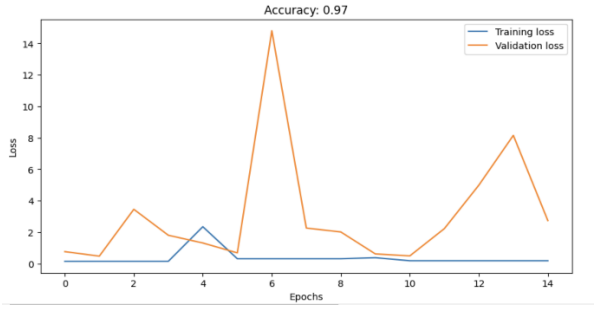
پروژه اول

مبانی هوش محاسباتی

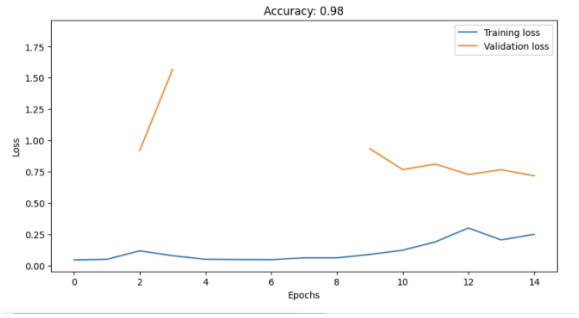
۱-انتخاب تعداد نورون هر لایه:

اول تعداد نورون یک لایه نهان را بررسی می‌کنیم:

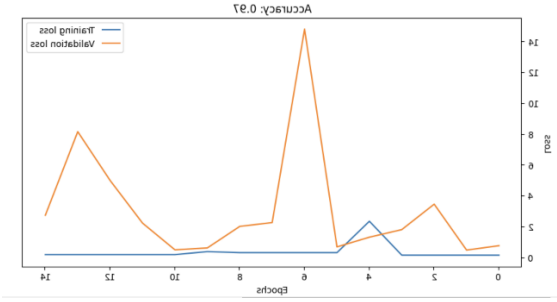
با مقادیر ۲۰۰، ۵۰۰، ۹۰۰ امتحان می‌کنیم، همانطور که مبینیم در حالت ۵۰۰ نورون بیشترین دقت را داریم. البته تنها با یکبار اجرا نمی‌توانیم چنین نتیجه‌ای بگیریم. همچنین خطای validation به مرور زمان کم شده، که این نشانه‌ی خوبی است.



۹۰۰ نورن



۵۰۰ نورون

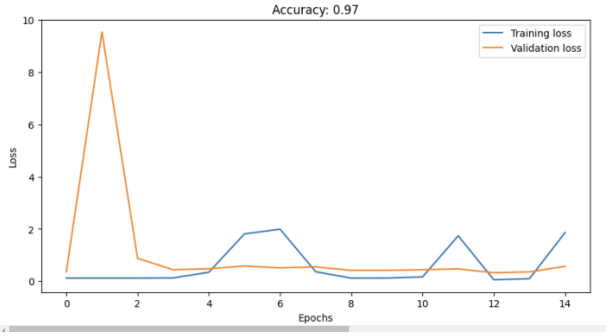


۲۰۰ نورون

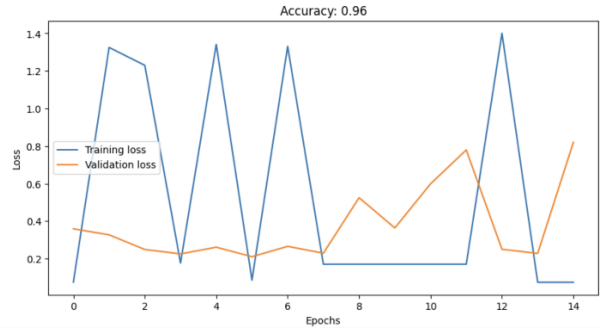
۲-انتخاب تعداد لایه‌ها:

حالت تک لایه با ۵۰۰ نورون را در مورد قبل مشاهده کردیم. حالا دو مورد دولایه‌ی ۵۰۰ نورونی و سه لایه ۵۰۰ نورونی را نیز مشاهده می‌کنیم.

در حالت دو لایه خطای validation کم‌تر از حالت‌های دیگر است. همچنین در حالت سه لایه علاوه بر خطای بیشتر، دقت هم کمتر است که می‌تواند نشانه‌ای بر رخ دادن بیش‌برازش باشد.



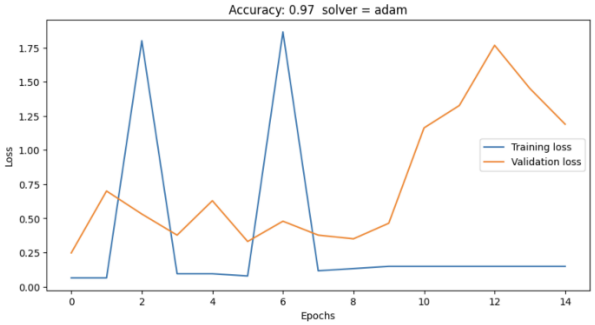
حالت دو لایه

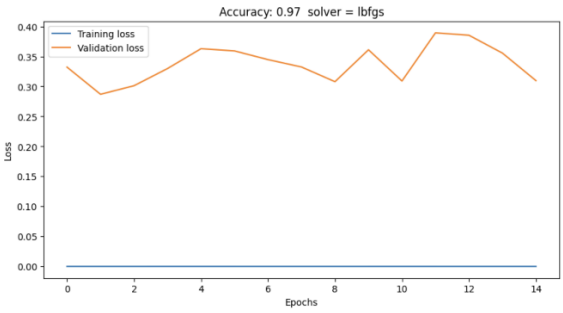


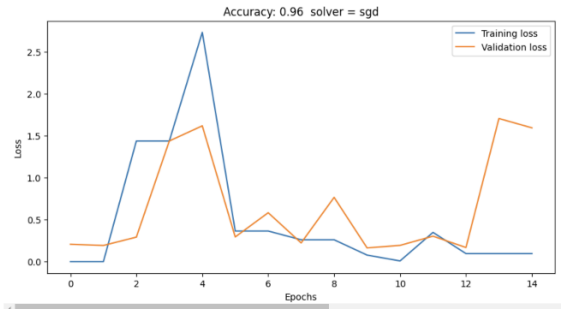
حالت سه لایه

۳_انتخاب الگوریتم بهینه‌سازی:

'lbfgs' یک بهینه ساز در خانواده روش های شبه نیوتنی است. 'sgd' به نزول گرادیان تصادفی اشاره دارد. 'adam'به یک بهینه ساز مبتنی بر گرادیان تصادفی اشاره دارد. در دیتاست‌های بزرگ، adam بهتر عمل می‌کند. و خطای validation در حال نزدیک شدن به خطای train است و پس از مدتی بیش‌برازش رخ می‌دهد.





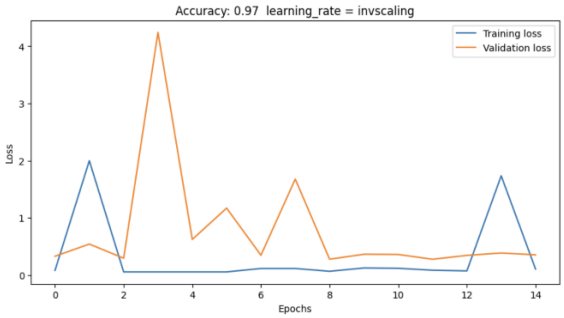


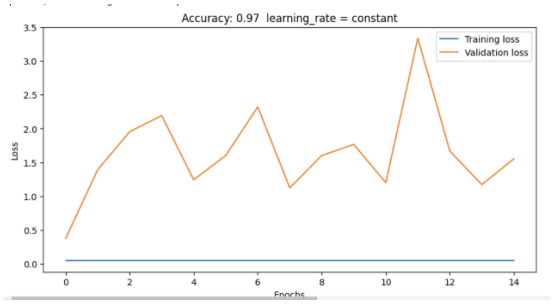
۴_انتخاب نرخ یادگیری:

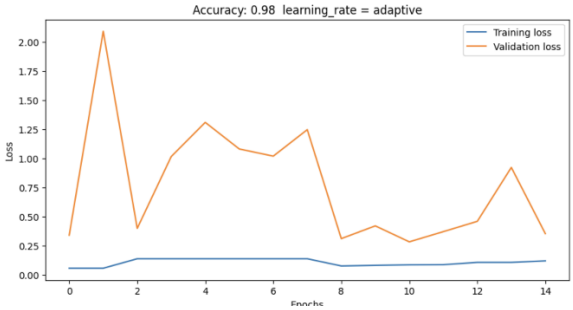
سه نوع انتخاب برای این گزینه وجود دارد:

یک نرخ یادگیری ثابت است که توسط «learning_rate_init» تعیین می‌شود. 'invscaling' به تدریج نرخ یادگیری را در هر مرحله زمانی 't' با استفاده از یک توان مقیاس معکوس 'power_t' کاهش می دهد. *'adaptive'* یا تطبیقی نرخ یادگیری را بر اساس loss تنظیم می‌کند و از این جهت هوشمندانه‌تر از دو روش دیگر است.

همانطور که می‌بینیم وقتی نرخ یادگیری ثابت است، خطای validation بالا میماند اما در دو حالت دیگر رو به کاهش است. قابل مشاهده است که درصد دقت نرخ یادگیری تطبیقی از دو روش دیگر بالاتر است.





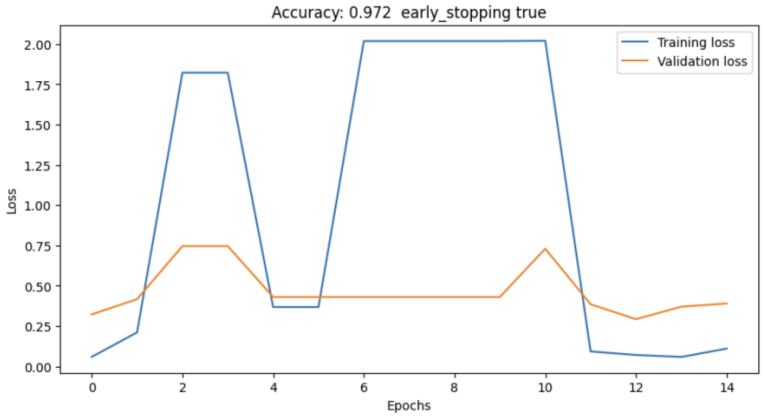


۵-تاثیر overfiting و underfitting :

هر دو دقت را کم کرده و از بهترین نتیجه دور می‌گرداند. از روی نمودارها، هر نقطه‌ای که قبل از آن خطای validation و خطای train هر دو نزولی هستند در حیطه‌ی underfitting قرار می‌گیرند و پس از آن نقطه و خطای train همچنان نزولی ولی خطای validation رو به افزایش است در محدوده‌ی overfiting یا بیش‌برازش قرار دارند. البته پیدا کردن مکان این نقطه به صورت کاملا دقیق نیست.

۶-شرایط توقف:

هنگامی که از early_stopping استفاده می‌کنیم، در صورتی که خطا در حال بهبود نباشد، آموزش را خاتمه می‌دهد که باعث جلوگیری از overfit می‌شود. (در حالت‌های قبلی این حالت غیر فعال بود.



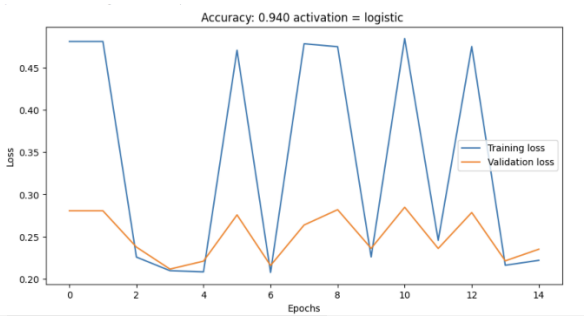
۷- تاثیر activation function :

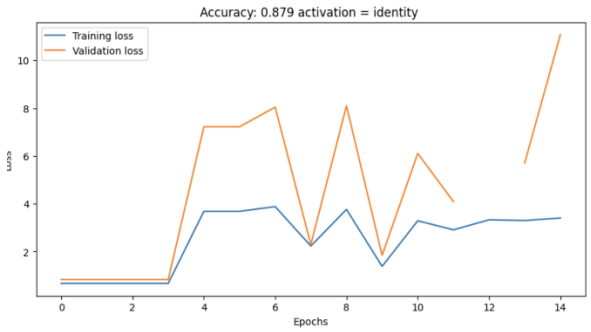
identity فعال‌سازی بدون عملیات، مفید برای پیاده‌سازی گلوگاه خطی، f(x) = x را برمی‌گرداند. دقت مدل پایین و خطای validation بالاتر از حالت‌های دیگر است.

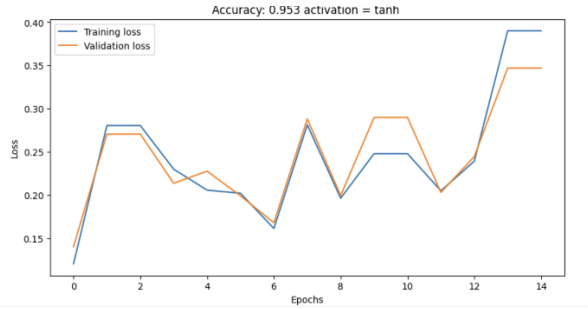
‘logistic’، تابع سیگموئید لجستیک، f(x) = 1 / (1 + exp(-x)) را برمی گرداند. دقت نسبت به حالت قبلی بهتر است اما نسبت به حالت اولیه کمتر است.

'tanh'، تابع هذلولی tan، f(x) = tanh(x) را برمی گرداند.حالت نسبت به حالت قبلی بهتر است ولی خطای بیشتری دارد.

'relu'، تابع واحد خطی اصلاح شده، f(x) = max(0, x) را برمی گرداند. (حالت در نمودارهای قبلی همین تابع بود، بنابراین دیگر این حالت را امتحان نمی‌کنیم.)



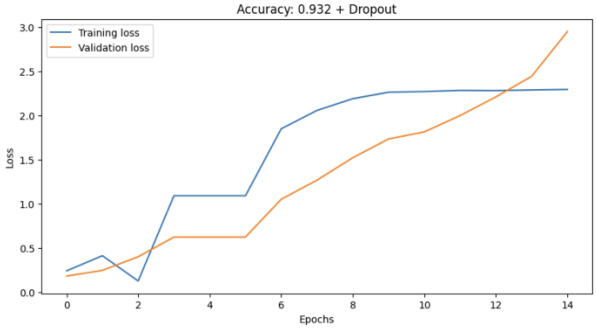




❄_تاثیر Dropout:

Dropoutیک تکنیک منظم‌سازی (regularization) است که برای کاهش بیش‌برازش (overfitting) در شبکه‌های عصبی استفاده می‌شود. این روش با حذف تصادفی برخی از نورون‌ها در هر لایه در طی فرآیند آموزش کار می‌کند. این کار باعث می‌شود که مدل وابستگی بیش از حد به نورون‌های خاص نداشته باشد و به این ترتیب یادگیری مدل بهبود پیدا می‌کند.

در نمودار می‌بینیم که بسیار زود به بیش‌برازش رسیدیم، این باعث میشود زودتر به پاسخ مطلوب برسیم.



❄_تاثیر Batch Normalization:

هدف آن بهبود سرعت و پایداری آموزش است. در نهایت اگر تعداد epoch ها کمتر بود دقت بالاتر می‌بود و زودتر به نقطه fit شدن می‌رسیم و در انتها بیش‌برازش رخ داده است.