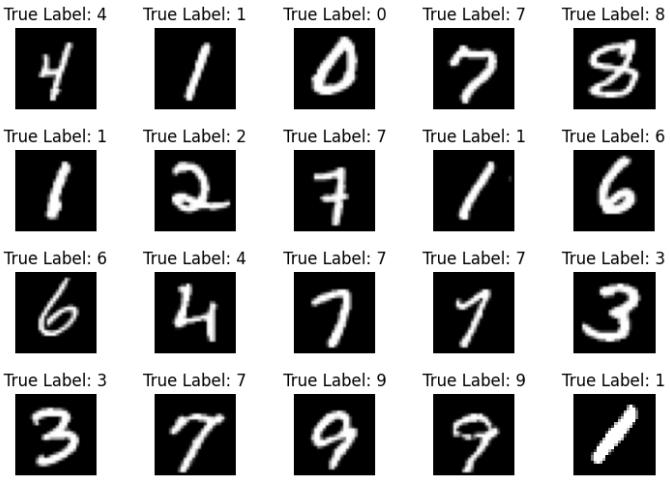
دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

آموزش مدل MLP بر روی داده‌های MNIST



سیدمحمدحسین هاشمی ۴۰۲۲۳۶۳۱۴۳

خرداد ۱۴۰۳

فهرست مطالب

چکیده

در دنیای هوش مصنوعی و یادگیری ماشین، شبکه‌های عصبی مصنوعی (ANNs) نقش بسیار مهمی در پردازش و تحلیل داده‌ها ایفا می‌کنند. یکی از معروف‌ترین و پرکاربردترین انواع این شبکه‌ها، شبکه عصبی پرسپترون چندلایه (MLP) است. MLP یکی از ساده‌ترین انواع شبکه‌های عصبی پیش‌خور[۱] محسوب می‌شود که شامل یک یا چند لایه مخفی[۲] بین لایه ورودی و خروجی است.

دیتاست MNIST[۳] یکی از این دیتاست‌هاست که به عنوان معیار استاندارد در بسیاری از پروژه‌های یادگیری ماشین و بینایی ماشین استفاده می‌شود. دیتاست MNIST شامل ۶۰۰۰۰ تصویر آموزشی و ۱۰۰۰۰ تصویر تست از ارقام دست‌نویس ۰ تا ۹ است. هر تصویر در این دیتاها به صورت یک ماتریس ۲۸x۲۸ پیکسل بوده و هر پیکسل مقداری بین ۰ تا ۲۵۵ را نشان می‌دهد که شدت رنگ خاکستری را نمایان می‌سازد.

آموزش یک MLP بر روی دیتاست MNIST به ما کمک می‌کند تا نحوه عملکرد این نوع شبکه‌ها را در تشخیص الگوهای پیچیده‌تر و طبقه‌بندی داده‌ها بهتر بفهمیم. در این فرآیند، مراحل مختلفی مانند پیش‌پردازش داده‌ها، طراحی و ساختاردهی شبکه، آموزش و ارزیابی مدل را باید طی کنیم.

۱ تعداد لایه‌های مخفی

در انتخاب تعداد لایه‌های مخفی[۴] برای شبکه عصبی پرسپترون چندلایه (MLP)، یکی از مهمترین ملاحظات، یافتن توازنی بین دقت آموزش و تست و همچنین پیچیدگی مدل است. نمودار ارائه شده نشان می‌دهد که چگونه تعداد لایه‌های مخفی بر دقت آموزش، دقت تست و تعداد تکرارهای لازم برای رسیدن به همگرایی تاثیر می‌گذارد. در شکل ۱ به مقایسه تعداد لایه‌ها در عملکرد شبکه پرداخته‌ایم.

دقت آموزش (Train Accuracy): با افزایش تعداد لایه‌های مخفی، دقت آموزش به طور کلی افزایش یافته و در حدود ۳۰ لایه به نقطه اشباع می‌رسد. این نشان می‌دهد که افزودن لایه‌های بیشتر بعد از این نقطه تاثیر قابل توجهی در بهبود دقت آموزش ندارد.

دقت تست (Test Accuracy): دقت تست نیز روند مشابهی را دنبال می‌کند و با افزایش تعداد لایه‌های مخفی بهبود می‌یابد تا به نقطه‌ای اشباع در حدود ۳۰ لایه برسد. بعد از این نقطه، دقت تست بهبود قابل توجهی نمی‌یابد.

تعداد تکرارها (Iterations): تعداد تکرارهای لازم برای همگرایی شبکه با افزایش تعداد لایه‌های مخفی کاهش می‌یابد. این نشان می‌دهد که پیچیدگی شبکه بعد از یک نقطه مشخص (حدود ۲۰ لایه) باعث کاهش زمان آموزش می‌شود و شبکه سریع‌تر به همگرایی می‌رسد. هرچند باید به این نکته نیز توجه داشت که تعداد لایه‌های بیشتر باعث می‌شود هرتکرار محاسبات بیشتری نیاز داشته باشد.

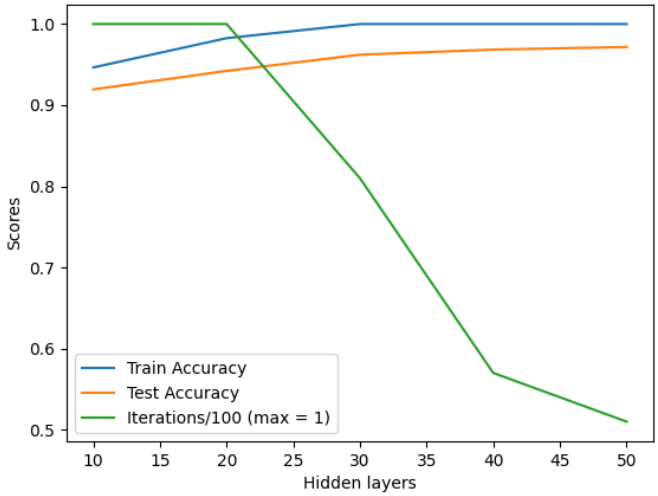
---

[1] feedforward

[2] hidden layers

[3] Modified National Institute of Standards and Technology

[4] Hidden Layers



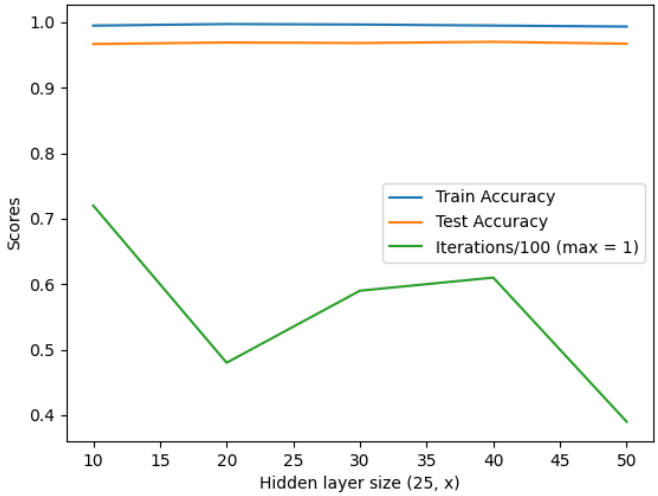
شکل ۱: تعداد لایه‌های مخفی

نتیجه‌گیری: با توجه به نمودار، به نظر می‌رسد که تعداد بهینه لایه‌های مخفی برای این شبکه در حدود ۳۰ - ۵۰ لایه است. در این محدوده، دقت آموزش و تست به بیشترین مقدار خود نزدیک شده و تعداد تکرارهای لازم برای همگرایی نیز در حداقل مقدار ممکن است. انتخاب تعداد لایه‌های بیشتر از این مقدار ممکن است منجر به پیچیدگی اضافی و زمان آموزش بیشتر بدون بهبود قابل توجه در دقت مدل شود.

توصیه‌ها: انتخاب بهینه تعداد لایه‌ها: بر اساس نمودار، ۵۰ لایه مخفی می‌تواند انتخاب مناسبی برای مدل MLP باشد.

اجتناب از بیش‌برازش (Overfitting): افزودن لایه‌های بیشتر می‌تواند به بیش‌برازش منجر شود که در آن مدل به خوبی بر روی داده‌های آموزش عمل می‌کند ولی دقت آن بر روی داده‌های تست کاهش می‌یابد. کارایی مدل: با توجه به پیچیدگی محاسباتی و زمان آموزش، انتخاب تعداد لایه‌های بهینه می‌تواند به افزایش کارایی و کاهش هزینه‌های محاسباتی کمک کند.

به این ترتیب، با توجه به تحلیل نمودار و بررسی دقت آموزش و تست و همچنین تعداد تکرارهای لازم برای همگرایی، انتخاب تعداد لایه‌های مخفی در حدود ۵۰ لایه به عنوان یک انتخاب بهینه پیشنهاد می‌شود.



شکل ۲: تعداد نورون‌ها در لایه‌های مخفی

۲ تعداد نورون‌های هرلایه

انتخاب تعداد نورون‌های هر لایه در شبکه عصبی پرسپترون چندلایه (MLP) یکی از مسائل مهم در طراحی شبکه است. هدف از این انتخاب، دستیابی به بالاترین دقت در آموزش و تست و همچنین کمترین تعداد تکرارهای لازم برای همگرایی است. نمودار ارائه شده، تاثیر تعداد نورون‌ها در هر لایه را بر دقت آموزش[۵]، دقت تست[۶]، و تعداد تکرارهای لازم برای همگرایی[۷] نشان می‌دهد. در شکل ۲ به مقایسه تعداد لایه‌ها در عملکرد شبکه پرداخته‌ایم.

تحلیل نمودار: دقت آموزش (Train Accuracy): با توجه به نمودار، دقت آموزش تقریباً ثابت و نزدیک به ۱ (۱۰۰ %) باقی می‌ماند که نشان می‌دهد مدل به خوبی قادر به یادگیری داده‌های آموزش است.

دقت تست (Test Accuracy): دقت تست نیز ثابت و نزدیک به ۱ (۱۰۰ %) باقی مانده و با تغییر تعداد نورون‌ها تفاوت چندانی ندارد، که نشان‌دهنده توانایی خوب مدل در تعمیم به داده‌های جدید است.

تعداد تکرارها (Iterations): تعداد تکرارهای لازم برای همگرایی شبکه با تغییر تعداد نورون‌ها در هر لایه تغییر می‌کند. کمترین تعداد تکرارها در حدود ۵۰ نورون در هر لایه به دست می‌آید، در حالی که با افزایش یا کاهش تعداد نورون‌ها، تعداد تکرارها ابتدا افزایش و سپس کاهش می‌یابد.

[5] Train Accuracy
[6] Test Accuracy
[7] Iterations

نتیجه‌گیری: با توجه به نمودار، به نظر می‌رسد که تعداد بهینه نورون‌ها در هر لایه حدود ۲۰ تا ۳۰ نورون است. در این محدوده، دقت آموزش و تست به حداکثر مقدار خود نزدیک بوده و تعداد تکرارهای لازم برای همگرایی نیز در کمترین مقدار خود است.

توصیه‌ها: انتخاب بهینه تعداد نورون‌ها: بر اساس نمودار، تعداد نورون‌های هر لایه در محدوده ۲۰ تا ۳۰ می‌تواند انتخاب مناسبی برای مدل MLP باشد.

اجتناب از پیچیدگی اضافی: افزایش یا کاهش شدید تعداد نورون‌ها می‌تواند منجر به افزایش پیچیدگی محاسباتی و زمان آموزش بدون بهبود قابل توجه در دقت مدل شود.

کارایی مدل: با انتخاب تعداد بهینه نورون‌ها، کارایی مدل افزایش یافته و هزینه‌های محاسباتی کاهش می‌یابد.

جمع‌بندی: با توجه به تحلیل نمودار و بررسی دقت آموزش و تست و همچنین تعداد تکرارهای لازم برای همگرایی، انتخاب تعداد نورون‌های هر لایه در محدوده ۲۰ تا ۳۰ به عنوان یک انتخاب بهینه پیشنهاد می‌شود. این انتخاب باعث می‌شود مدل با دقت بالا و زمان آموزش کمتر به همگرایی برسد و عملکرد مناسبی در طبقه‌بندی داده‌ها داشته باشد.

به این ترتیب، با توجه به دقت آموزش و تست بالا و تعداد تکرارهای کم، می‌توان گفت که انتخاب تعداد نورون‌های هر لایه در محدوده ۲۰ تا ۳۰ نورون بهینه است و می‌تواند بهترین نتیجه را در آموزش مدل MLP بر روی دیتابیس MNIST به همراه داشته باشد.

۳ توابع بهینه سازی

در کتابخانه scikit-learn[8] در Python، توابع بهینه سازی که برای آموزش مدل های MLP[9] استفاده می‌شوند عبارتند از:

۱.۳ تابع بهینه‌سازی SGD

- SGD[10] یک الگوریتم بهینه سازی اول مرتبه است که برای آموزش مدل های ماشین یادگیری استفاده می شود.
- در این روش، به جای استفاده از گرادیان کل داده ها، از نمونه های تصادفی از داده ها برای محاسبه گرادیان استفاده می شود.
- این روش محاسبات را ساده تر و کارآمدتر می کند و برای مقیاس پذیری مناسب است.
- SGD معمولاً برای مدل های بزرگ و داده های با حجم زیاد مناسب است.

[8] sklearn
[9] Multi-Layer Perceptron
[10] Stochastic Gradient Descent

۲.۳ تابع بهینه‌سازی L-BFGS

- L-BFGS[11] یک الگوریتم بهینه سازی مرتبه دوم است که از تقریب ماتریس هسین استفاده می کند.
- این روش به جای محاسبه مستقیم ماتریس هسین، از یک ساختار داده ای کم حافظه برای نگهداری اطلاعات مربوط به ماتریس هسین استفاده می کند.
- L-BFGS همگرایی سریع تری نسبت به SGD دارد و برای مسائل با فضای پارامتری کوچک تر مناسب است.
- این روش محاسبات پیچیده تری نسبت به SGD دارد، اما همچنان در مقایسه با روش های مرتبه دوم سنتی کارآمد است.

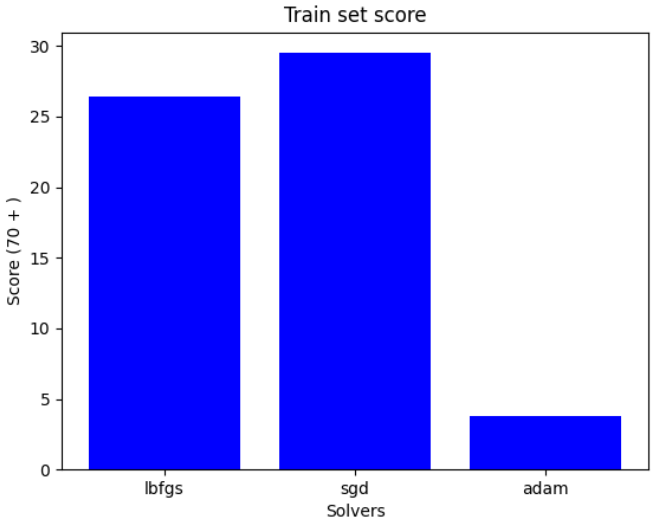
۳.۳ تابع بهینه‌سازی Adam

- Adam[12] یک الگوریتم بهینه سازی است که از محاسبات گرادیان تطبیقی برای آموزش مدل های ماشین یادگیری استفاده می کند.
- در این روش، به جای تنظیم نرخ یادگیری به صورت ثابت، نرخ یادگیری به صورت تطبیقی برای هر پارامتر بروز رسانی می شود.
- Adam از میانگین متحرک گرادیان و مربع گرادیان برای تنظیم نرخ یادگیری استفاده می کند.
- Adam در مقایسه با روش های بهینه سازی سنتی مانند SGD، همگرایی سریع تری دارد و برای مسائل پیچیده مناسب است.

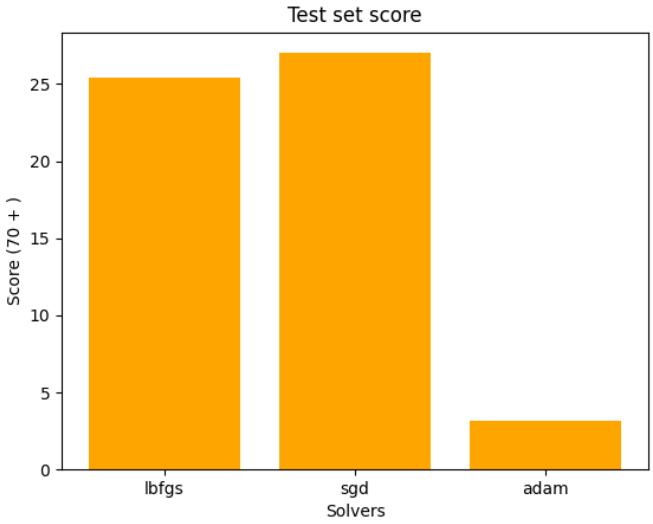
۴.۳ مقایسه سه روش

نمودار مقایسه این سه تابع بهینه سازی را در تصاویر ۳، ۴ و ۵ و نتیجه مقایسه این سه روش را در ۱ مشاهده می‌کنید.

[11] Limited-memory Broyden-Fletcher-Goldfarb-Shanno
[12] Adaptive Moment Estimation



شکل ۳: مقایسه دقت آموزش در توابع



شکل ۴: مقایسه دقت تست در توابع

شکل ۵: مقایسه تعداد تکرار در توابع

جدول ۱: مقایسه توابع بهینه سازی

تابع بهینه سازی	تعداد تکرار	دقت تست	دقت آموزش
L-BFGS	بالا	بالا	بالا
SGD	پایین	بالا	بالا
Adam	پایین	پایین	پایین

نتیجه گیری: طبق نتایج دریافت شده واضح است که استفاده از تابع بهینه‌سازی SGD علاوه بر دقت بالا در آموزش و تست به تعداد تکرار کمتری نیز برای آموزش نیاز دارد.

۴ توابع فعال سازی

توابع فعال‌سازی در شبکه‌های عصبی یکی از اجزای کلیدی هستند که نقش مهمی در عملکرد و یادگیری مدل‌ها دارند. در کتابخانه Scikit-Learn نیز توابع فعال‌سازی مختلفی برای استفاده در مدل‌های عصبی وجود دارند. در ادامه، به توضیح مختصری از تعدادی از توابع فعال‌سازی موجود در Scikit-Learn

می‌پردازیم:

۱.۴ Identity

- این تابع هیچ تغییری در ورودی ایجاد نمی‌کند و خروجی همان ورودی است.
- فرمول آن به صورت زیر است:

$$f(x) = x$$

۲.۴ ReLU (Rectified Linear Unit)

- این تابع برای حذف خطی‌تر کردن خروجی نورون‌ها استفاده می‌شود.
- فرمول آن به صورت زیر است:

$$f(x) = \max(0, x)$$

۳.۴ Logistic (Sigmoid)

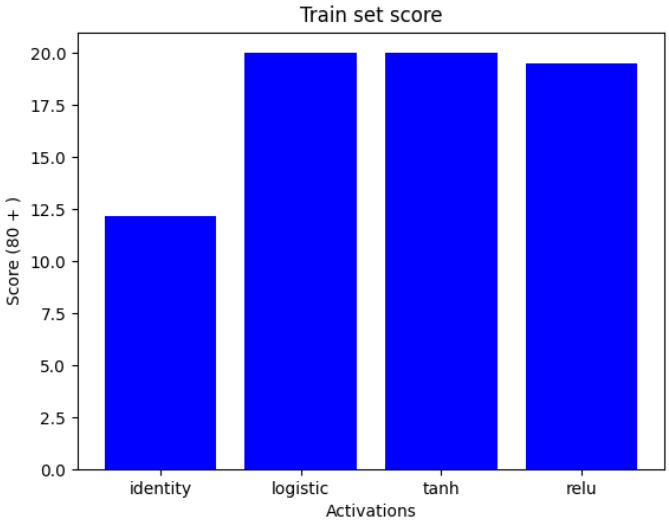
- این تابع برای مسائل طبقه‌بندی دو کلاسه استفاده می‌شود.
- فرمول آن به صورت زیر است:

$$f(x) = \frac{1}{1+e^{-x}}$$

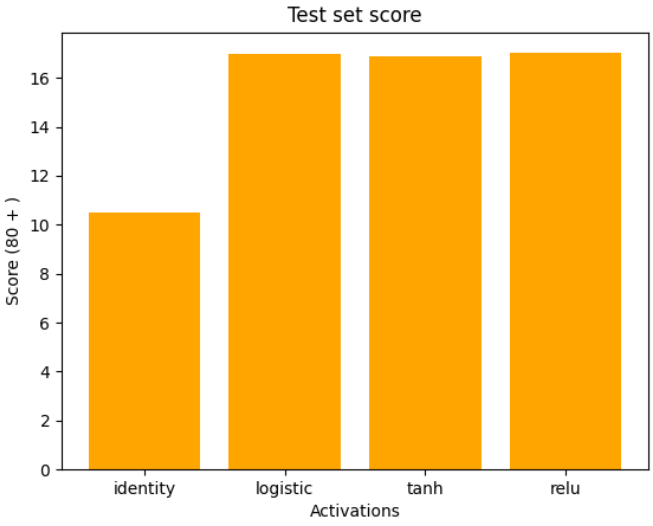
۴.۴ Tanh (Hyperbolic Tangent)

- این تابع مقادیر خروجی را در بازه [-1, 1] قرار می‌دهد.
- فرمول آن به صورت زیر است:

$$f(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$$



شکل ۶: مقایسه دقت آموزش در توابع



شکل ۷: مقایسه دقت تست در توابع

شکل ۸: مقایسه تعداد تکرار در توابع

مقایسه: مدل را با استفاده از هر سه تابع آموزش داده و نتایج را باهم مقایسه می‌کنیم. نمودار های ۶، ۷ و ۸ نمودار مقایسه توابع فعال سازی و جدول ۲ نتیجه مقایسه را نشان می‌دهند.

جدول ۲: مقایسه توابع بهینه سازی

تابع فعال سازی	تعداد تکرار	دقت تست	دقت آموزش
identity	متوسط	متوسط	متوسط
logistic	بالا	بالا	بالا
tanh	پایین	بالا	بالا
relu	متوسط	بالا	بالا

نتیجه‌گیری: استفاده از tanh به نسبت سایر توابع فعال سازی، علاوه بر دقت بالا در آموزش و تست، به تعداد تکرار کمتری برای آموزش نیاز دارد.

۵ نرخ یادگیری

نرخ یادگیری[۱۳] در شبکه‌های عصبی یک پارامتر بسیار مهم است. این مقدار عددی که در آموزش مدل با روش کاهش شیب[۱۴] استفاده می‌شود، تعیین می‌کند که مدل چقدر سریع یاد می‌گیرد. در هر گام، الگوریتم کاهش شیب مقدار نرخ یادگیری را در گرادیان‌ها یا شیب‌ها ضرب می‌کند. حاصل ضرب این‌ها گام شیب نامیده می‌شود.

نرخ یادگیری یک ابرپارامتر[۱۵] کلیدی است. انتخاب مناسب نرخ یادگیری می‌تواند تأثیر زیادی بر عملکرد مدل داشته باشد. اگر نرخ یادگیری بزرگ باشد، ممکن است مدل به سرعت همگرا شود، اما به مشکلاتی مانند شلوغی یا نوسان‌های ناپایدار برخورد کند. اگر نرخ یادگیری کوچک باشد، مدل به طور کامل همگرا نشود یا به سرعت در مینیمم محلی گیر کند.

برای تعیین نرخ یادگیری مناسب، معمولاً از روش‌های آزمون و خطا، تجربی، و تنظیم دستی استفاده می‌شود. همچنین الگوریتم‌های پیشرفته‌تری نیز برای تعیین نرخ یادگیری به کار می‌روند.

در کل، انتخاب نرخ یادگیری به تجربه و آزمون‌های مکرر بر مدل‌ها برمی‌گردد تا بهترین مقدار برای مسئله‌ی خاص مشخص شود.

در اینجا برای دریافت نرخ یادگیری نموداری مقایسه ای برای مقادیر مختلف را محاسبه کرده‌ایم که این نمودار را در شکل ۹ مشاهده می‌کنید.

نتیجه‌گیری: طبق نمودار پس از نرخ یادگیری 0.55 دقت مدل در آموزش و تست بالای ۹۰% شده ولی تعداد تکرار بسیار بالاست ولی در مقدار 0.11 علاوه‌بر دقت بالا در آموزش و تست، تکرار کمتری برای آموزش نیاز دارد که در نتیجه برای نرخ یادگیری از مقدار 0.1 تا 0.2 استفاده می‌کنیم.

۶ اوورفیتینگ و آندرفیتینگ

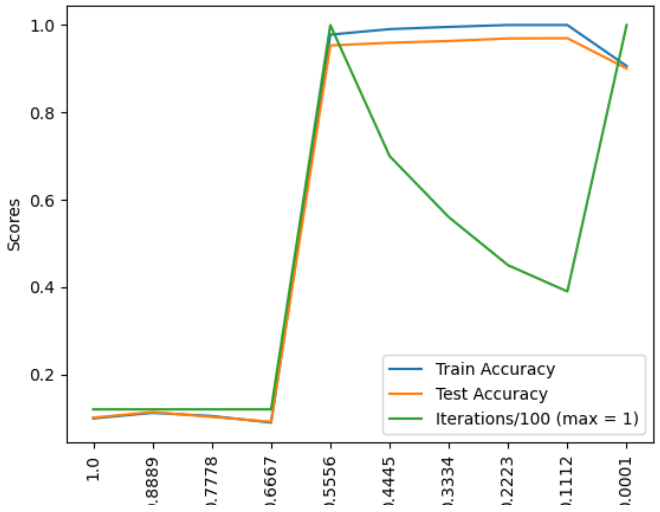
آندرفیتینگ[۱۶]: وقتی مدل خیلی ساده‌ای است و نمی‌تواند الگوهای مهم در داده‌ها را یاد بگیرد، آن را بازتولید کند و یا پیش‌بینی دقیقی انجام دهد، ما آن را آندرفیتینگ می‌نامیم. به عبارت دیگر، مدل آندرفیت شده است و از توانایی کلیه الگوهای موجود در داده‌ها به درستی استفاده نمی‌کند.

اوورفیتینگ[۱۷]: وقتی مدل خیلی خود را به داده‌های آموزش برازش می‌کند و الگوهای موجود در داده‌های آموزش را به اندازه‌ی کافی یاد می‌گیرد، اما نتوانسته الگوهای کلی را درک کند و به‌طور کلی بر روی داده‌های جدید عمل نمی‌کند، ما این پدیده را اوورفیتینگ می‌نامیم. به عبارت دیگر، مدل اوورفیت شده است و به داده‌های آموزش بسیار حساس است و به داده‌های جدید عملکرد بدی دارد.

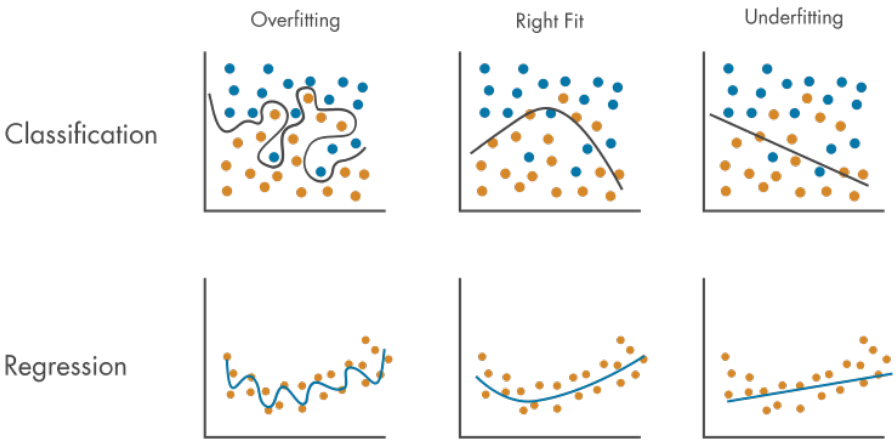
آندرفیتینگ و اوورفیتینگ در شکل ۱۰ مشاهده می‌کنید.

---

[13] Learning Rate
[14] gradient descent
[15] hyperparameter
[16] Underfitting
[17] Overfitting



شکل ۹: تاثیر نرخ یادگیری در مدل



شکل ۱۰: اوورفیتینگ و آندرفیتینگ

۷ شرایط توقف

در مدل MLP در کتابخانه scikit-learn، شرایط توقف مربوط به چندین پارامتر مختلف می‌تواند تعیین شود:

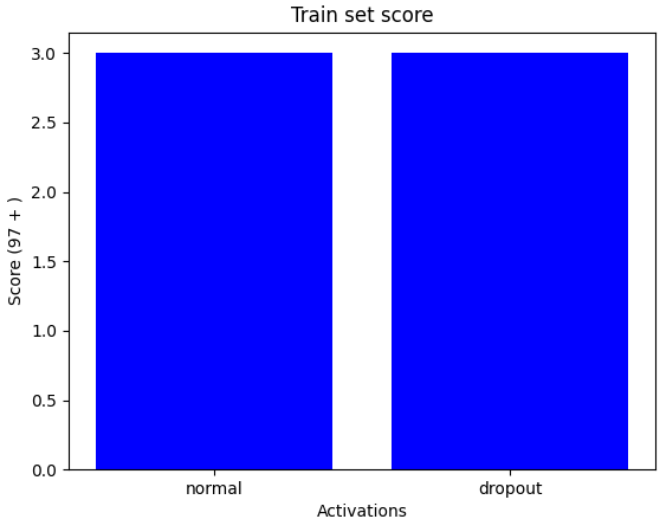
- max_iter: این پارامتر تعیین می‌کند که آموزش مدل تا چند ایپوک ادامه یابد. اگر تعداد ایپوک‌های آموزش به max_iter برسد و مدل هنوز هم به شرایط توقف نرسیده باشد، آموزش متوقف می‌شود.
- tol: این پارامتر مقدار تعیین‌کننده‌ای است که در صورتی که تغییرات وزن‌های مدل در طول آموزش به آن‌ها کمتر از مقدار مشخص شده باشد، آموزش متوقف می‌شود.
- early_stopping: این پارامتر مشخص می‌کند که آیا از شرایط توقف زودهنگام استفاده شود یا خیر. اگر مقدار آن True باشد، مدل در صورتی که بهترین عملکرد روی داده‌های اعتبارسنجی بهبود نکند، آموزش متوقف می‌شود.
- early_stopping_patience: این پارامتر تعیین می‌کند که مدل به مدت چند ایپوک بدون بهبود عملکرد روی داده‌های اعتبارسنجی آموزش را ادامه دهد قبل از اینکه تصمیم به توقف بگیرد.
- n_iter_no_change: این پارامتر تعیین می‌کند که مدل به مدت چند ایپوک بدون بهبود در معیار عملکرد، آموزش را متوقف کند. این پارامتر در صورتی که early_stopping فعال باشد، معنا پیدا می‌کند.

با استفاده از ترکیب این پارامترها، می‌توانید شرایط توقف مدل MLP را با دقت و کنترل بیشتری مدیریت کنید.

۸ تاثیر dropout

Dropout یکی از روش‌های مهم برای کاهش اوورفیتینگ در شبکه‌های عصبی است. در این روش، به صورت تصادفی برخی از واحدهای (نورون‌ها) در شبکه در هنگام آموزش غیرفعال می‌شوند. به این ترتیب، هر بار که داده از شبکه عبور می‌کند، یک زیرمجموعه مختلف از واحدها غیرفعال می‌شوند و شبکه با اینکه به داده‌های آموزشی تطبیق می‌یابد، از یادگیری وابسته به ویژگی‌های خاص داده‌ها جلوگیری می‌کند.
تاثیر Dropout در شبکه‌های عصبی به‌طور خلاصه به شرح زیر است:

۱. کاهش اوورفیتینگ: Dropout از این اتفاق جلوگیری می‌کند که شبکه به طور زیادی به داده‌های آموزشی برازش شود و الگوهای آموزشی را حفظ کند که ممکن است برای داده‌های تست مناسب نباشد.



شکل ۱۱: تاثیر dropout در آموزش

۲. افزایش عمومیت[۱۸]: با غیرفعال کردن بخشی از واحدها، شبکه نه تنها به داده‌های آموزشی بلکه به الگوهای کلی و عمومی تری اعمال می‌شود. این باعث می‌شود که شبکه قادر به تعمیم‌پذیری بهتری بر روی داده‌های تست باشد.

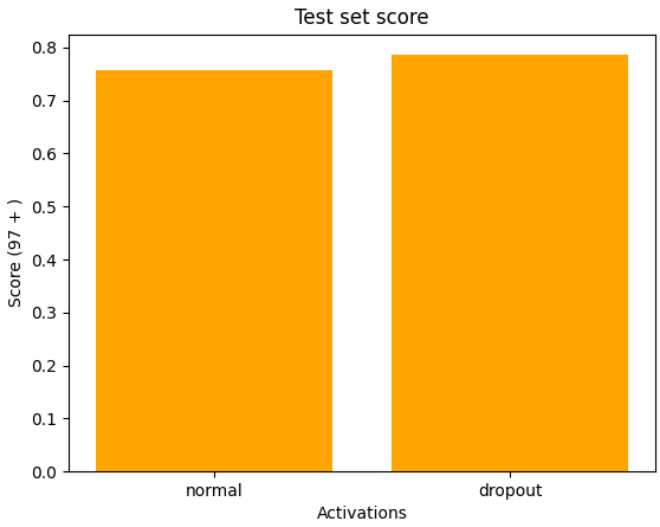
۳. کاهش وابستگی به ویژگی‌های خاص: با غیرفعال کردن بخشی از واحدها، شبکه به یادگیری از ویژگی‌های خاص داده‌ها کمتر وابسته می‌شود و به جای آن الگوهای کلی‌تر و عمومی‌تری را یاد می‌گیرد.

با این وجود، استفاده از Dropout نیازمند تنظیمات صحیح است و اعمال Dropout به طور نادرست ممکن است عملکرد شبکه را به طرز منفی تحت تأثیر قرار دهد.
در نمودارهای ۱۲، ۱۱ و ۱۳ می‌توانیم تاثیر Dropout را مشاهده کنید.

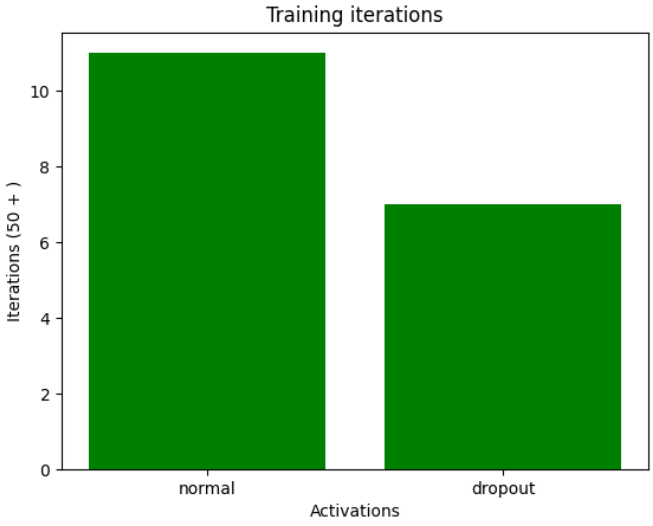
۹ تاثیر batch_size

انتخاب مناسب batch_size در آموزش شبکه‌های عصبی تأثیر زیادی بر عملکرد و سرعت آموزش دارد. batch_size تعیین می‌کند که در هر مرحله چه تعداد نمونه از داده‌ها برای محاسبه خطا و

[18] Generalization



شکل ۱۲: تاثیر dropout در تست



شکل ۱۳: تاثیر dropout در تعداد تکرار

به‌روزرسانی وزن‌ها استفاده شود. این پارامتر تاثیرات متعددی دارد که به طور خلاصه به شرح زیر است:

- سرعت آموزش: انتخاب batch_size بزرگتر می‌تواند منجر به سرعت بیشتری در آموزش شود. با استفاده از بچ‌های بزرگتر، محاسبات بیشتری همزمان انجام می‌شود که باعث می‌شود فرآیند آموزش سریع‌تر انجام شود.
- پایداری آموزش: با انتخاب batch_size کوچکتر، یعنی استفاده از بچ‌های کوچک‌تر، ممکن است عملکرد شبکه در حین آموزش پایدارتر باشد. این به این معنی است که وزن‌ها به طور متوسطتر به‌روزرسانی می‌شوند و در نتیجه آموزش به سمت یک مینیمم محلی نرفته و از مینیمم‌های محلی دیگری نیز گریزان می‌شود.
- حافظه مصرفی: انتخاب batch_size بزرگتر می‌تواند حافظه مصرفی را به طور قابل توجهی کاهش دهد، زیرا تعداد بیشتری از نمونه‌ها در هر مرحله آموزش بارگذاری می‌شوند.
- دقت آموزش: انتخاب مناسب batch_size می‌تواند به دقت آموزش کمک کند. بچ‌های بزرگتر ممکن است باعث شود که شبکه به سرعت به مینیمم محلی برسد و در نتیجه دقت کمتری داشته باشد، در حالی که بچ‌های کوچکتر ممکن است به دقت آموزش کمک کنند.

بنابراین، انتخاب batch_size باید با توجه به مسئله و محیط آموزش انجام شود و نیاز به آزمون و خطا برای انتخاب بهترین مقدار دارد.

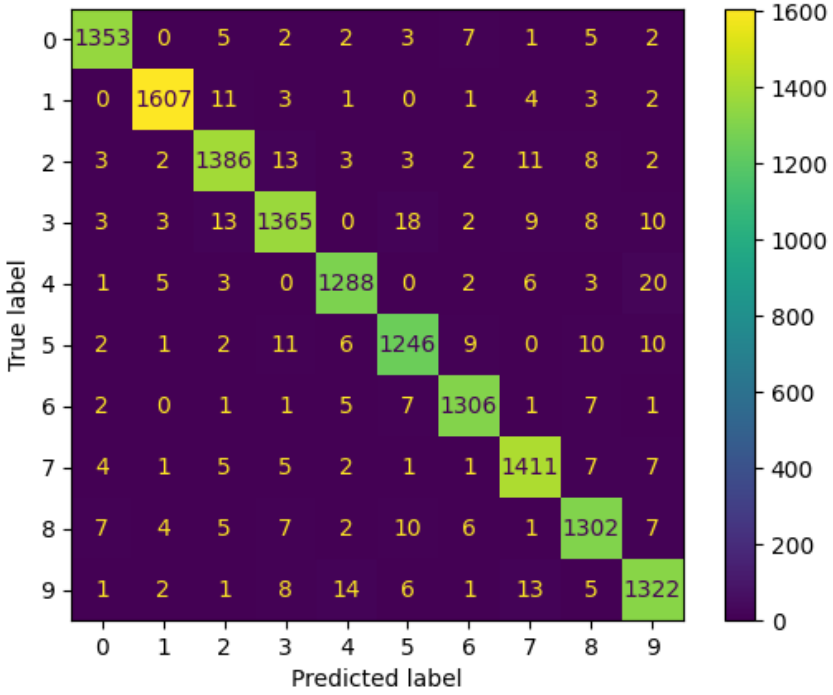
در شکل ۱۴ تاثیر batch_size را در آموزش، تست و تعداد تکرار مشاهده می‌کنیم.

نتیجه‌گیری: batch_size در مقادیر ۲۵۰ تا ۳۵۰ علاوه‌بر افزایش دقت آموزش و تست، تعداد تکرار را نیز کاهش می‌دهد.

۱۰ مدل نهایی

در نهایت مدل نهایی با confusion matrix شکل ۱۵ به دست می‌آید.

شکل ۱۴: تاثیر batch_size در مدل



شکل ۱۵: confusion matrix مدل نهایی