جلسه دوم:

جلسه سوم:

اگر تابع فعالیت یکنوا افزایشی باشد به این صورت کاری میکنیم که خروجی ما بیشتر بشود یعنی اگر خروجی درست ما 1 بوده و خروجی ما کمتر بوده مثلا -1 کمک میکند بیشتر شود و همگرا میشود و بدین ترتیب تغییر وزن ها باعث کاهش خطا میشود. فرض جدایی پذیر خطی مهم هست در استفاده از پرسپترون. اگر وزن بهینه را در بیاوریم یعنی معادله خط یا مرز تصمیم گیری را بدست آورده ایم حالا بستگی به بعد هم دارد.

نحوه محاسبه مثال:

وزن را که ابتدا صفر قرار دادیم نظیر به نظیر با ورودی ضرب میکنیم از تابع فعالیت عبور میدهیم اگر خروجی با خروجی واقعی یکی بود که هیچی اگر نبود اپدیت میکنیم، چطوری؟ ورودی را در خروجی واقعی ضرب میکنیم بعلاوه یا منهای وزن فعلی میکنیم.

قضیه همگرا اگر مسئله جدایی پذیر خطی باشد در صورت وجود وزن ها مسئله به همگرایی میرسد و پاسخ همگرا خواهد بود.

Adaline تابع خروجی آن تابع خطی است چرا؟ در LMS میگفت میزان مقدار خطا مهم هست و اگر خطا بالا بود اپدیت بیشتر باشد بر خلاف پرسپترون که میزان خطا تاثیری نداشت و در نتیجه در LMS همگرایی سریع تر هست. Adaline تابع خطی دارد دیگر -1 و +1 نیست چرا؟ وقتی میخواهد تاثیر خطا در نظر بگیرد دیگر تابع نباید دو سطحی یا این یا اون باشد باید میزان خطا را در نظر بگیرد پس خروجی خطی هست. در Adaline از یک تابع فعالیت خطی رد میشود با وجود اینکه خروجی آن دو سطحی هست تنها نکته اینجا هست که بر خلاف پرسپترون اینجا مهم میشود که چه قدر خطا داریم و فقط داشتن یا نداشتن خطا مهم نیست.

جلسه چهارم:

LMS برخلاف پرسپترون میزان خطا در آن مهم هست در صورتی که در پرسپترون صرفا اگر خطا داشت میگفتی کم کن یا زیاد کن. در نتیجه همگرایی زودتر است. واسه همین هست که ما از تابع خطی استفاده میکنیم و feedback میدهیم. بعد حالا به صورت دو سطحی ها غیر خطی خروجی میدهیم. در واقع اولین تابع فعالیت ما خطی هست بر خلاف پرسپترون تا بتوانیم فیدبک بدهیم. نرخ آموزش بزرگی تغییرات وزن ها را مشخص میکند اگر زیاد باشد یعنی میزان تغییر وزن باید زیاد باشد.

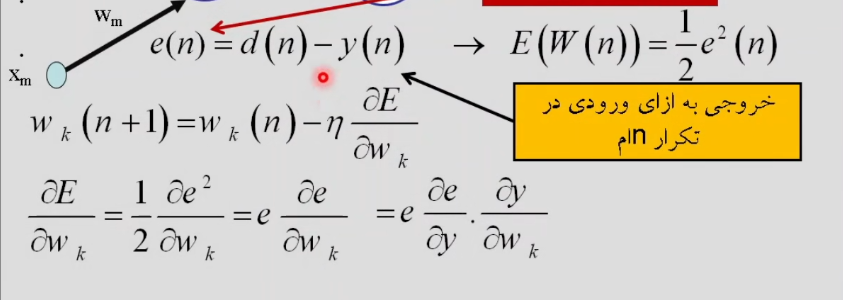
Epoch: یکبار همه ورودی ها را ببینی و خطا را حساب بکنی و MSE آن را بدست بیاوری بعد وزن ها را اپدیت بکنی، میشود یک دوره. البته batch گفته میشود به این سیستم که همه ورودی ها را ببینی و خطا را حساب بکنی و اپدیت وزن انجام بدهی که در مخالف sequence mode هست.

وزن ها پارامتر های آزاد و قابل تغییر ما هستند برای تابع خطا چون با تغییر وزن میتوانیم کاهش خطا بدهیم.

یک وزنی در نظر میگیریم تابع خطا را حساب میکنیم در عکس گرادیان یا همان مشتق وزن را بروز رسانی میکنیم یعنی اگر شیب منفی بود یا گرادیان منفی بود وزن را زیاد میکنیم و اگر شیب مثبت بود و گرادیان مثبت بود وزن را کم میکنیم. به این میگوییم steepest descent. اگر بیشتر از یک متغیر بود بردار گرادیان داریم.

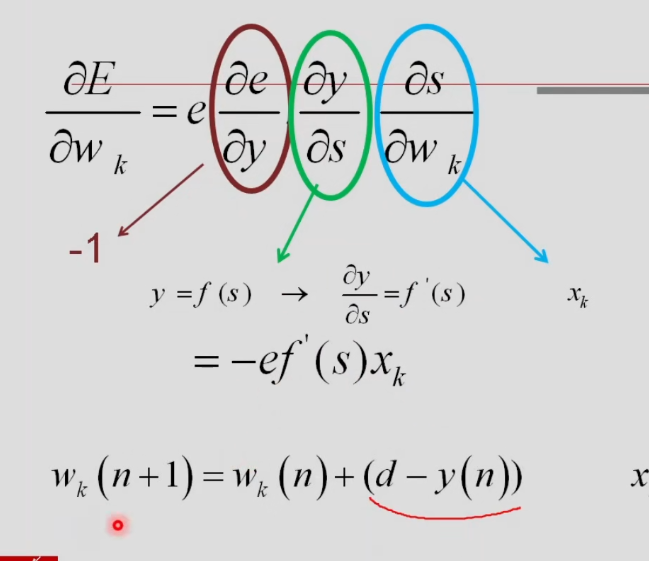
در نرخ یادگیری کم همگرایی کند و نرخ یادگیری زیاد ناپایدار میشود سیستم.

Sequential یعنی ورودی ها دونه به دونه میان و خطا حساب میشود و وزن اپدیت میشود. روند اپدیت شدن تابع خطا روند نرم تری در batch mode داریم ولی در sequential تابع ما زیاد نرم نیست. چون batch دارد میانگین خطا میگیرد دیگه.



در اینجا از مشتق زنجیره ای استفاده میکنیم بخاطر اینکه ما مشتق ارور را به صورت جزئی میخواهیم از وزن بگیریم اما مستقیم به آن دسترسی نداریم بلکه در y قرار دارد پس باید زنجیره ای بریم تا به w برسیم.

دقت کن d ثابت هست در حین مشتق گیری صفر میشود.



طبق عکس بالا واسه همین هست که مشتق تابع سیگموید در حین محاسبه خطا نیاز داریم.

پرسپترون 3 ورودی مرز جدا سازش صفحه میشود.