گزارش تمرین سری اول

هدف ازین تمرین یافتن احتمال قبولی یا رد شدن دانشجویان بر اساس پارامتر هایی نظیر توجه در کلاس، ساعت تمرین کردن، استعداد و غیره است.

اطلاعات تعدادی از دانشجویان و نتیجه ی آنان در یک فایل csv. جمع آوری شده که از انان به عنوان داده ی اولیه استفاده میکنیم.

روش کار بدین صورت است که ابتدا ما باید داده ها را با تابع getdata از فایل csv. بخوانیم و ان ها را در یک بردار دو بعدی ذخیره کنیم.

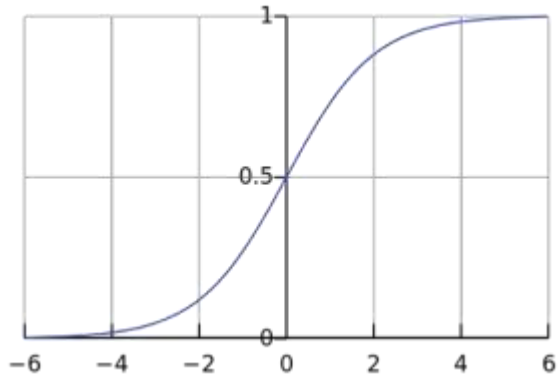
در این تابع یک پارامتر bias قرار میدهیم که در صورت true بودن به اول بردار اطلاعات هر دانشجو عدد ۱ را برای بایاس کردن اضافه میکند.

این تابع بردار دو بعدی dataList که به تعداد دانشجویان دارای سطر و در هر سطر اطلاعات هر دانشجو قرار دارد برمیگرداند.

سپس میتوانیم داده ها را با تابع displayDataset نمایش دهیم. برای زیبایی برای هر ستون طول مشخصی در نظر میگیریم و alignment همه ی آن ها برای منظم بودن left قرار میدهیم.

جدول ذکر شده را در دو حالت باید نمایش دهیم. حالت اول زمانی که بایاس داریم و حالت دوم زمانیکه بایاس نداشته باشیم.

حال باید تابعی تعریف کنیم که به کمک خروجی آن بتوانیم تشخیص دهیم دانشجو قبول یا رد میشود. برای این کار از تابع hypothesis استفاده میکنیم. خروجی این تابع بین ۰ و ۱ میباشد و حالت بایاس آن به ازای وزن های صفر روی نقطه ی ۰/۵ قرار دارد. هدف ما به دست اوردن بهترین بردار وزن است به گونه ای که اگر دانشجو پاس شده باشد خروجی این تابه به ۱ نزدیک باشد و اگر دانشجو افتاده باشد خروجی این تابع به صفر نزدیک باشد. هر چقدر این تابع به ۰ و ۱ نزدیکتر جواب دهد یعنی روند طراحی ما دقیقتر بوده است.



نمودار تابع hypothesis

ابتدا تابع h را با استفاده از فرمول ذکر شده در توضیحات تمرین مینویسیم. سپس برای یافتن بهترین بردار وزن باید تابع cost function را به گونه ای طراحی کنیم که min شود سپس بردار وزن در این حالت بردار مطلوب میشود.

تابع J با توجه به رابطه ی موجود در توضیحات تمرین مینویسیم. برای یافتن بهترین وزن و مینیمم کردن تابع J از تابع fitOneEpoch و fit استفاده میکنیم.

این تابع داده ها را به دسته های کوچکتر تقسیم میکند و از بردار اولیه وزن به عنوان نقطه ی شروع استفاده میکند و در هر iteration تابع وزن را اپدیت میکند. این کار را به ازای همه ی دسته ها فقط یکبار انجام میدهد.

سپس با استفاده از تابع fit این کار را برای تعداد مشخصی iteration انجام میدهیم تا بتوانیم تابع وزن دلخواه را پیدا کنیم.

اگر در این حالت توابع نوشته شده درست عمل کنند به ازای هر epoch (یکبار صدا زدن تابع fitOneEpoch) تابع cost function باید به سمت صفر همگرا شود.

بنابراین به راحتی میتوانیم بردار وزن را محاسبه کنیم.

سپس داده های تست را با بردار وزن به تابع predict میدهیم و با استفاده از خروجی تابع h میتوانیم احتمال پاس شدن و قبول شدن دانشجو را پیدا کنیم.