جلسه اول:

الگوریتم یادگیری میاد به فضای فرضیه نگاه میکند و یکی از آنها را که سازگار با داده های آموزشی هست انتخاب میکند. فرضیه همان فانکشن هدف ما هست که ورودی را به خروجی نگاشت میکند.

پرسپترون یک رابطه خطی مدل میکند. پرسپترون به مقادیر ورودی یک وزنی اختصاص میدهد و میگوید از این آستانه بالاتر بود فلان اگر نبود فلان. مثلا وام بدهیم یا ندهیم. آستانه عضو فضای فرضیه هست.

اگر داده های ما جدایی پذیر خطی باشند پرسپترون در تعداد متنهای میتواند این داده ها را از هم جدا کند.

در یادگیری غیر نظارتی به صورت ذاتی میتوانیم داده ها را با خطوط به خوشه ها تبدیل کنیم که تجمع داده در آنجا زیاد است.

موارد قبلی همه به صورت اپیزودیک بود و مقطعی بود یعنی این مشتری خوب هست یا نیست، اما در یادگیری تقویتی ما یک سری گام داریم و خروجی ما ترتیبی و بر اساس یک سلسله تصمیمات هستش. اینجا مسئله تصمیم گیری دنباله دار هست و تصمیمات فعلی در آینده هم تاثیر گذار است. و یک میزان مطلوبیت برای خروجی خودمان داریم و هدف یک پالیسی هستش که به ازای هر ورودی چه اکشنی را انجام بدهیم تا به مطلوبیت برسیم.

گاهی اوقات ممکن است یادگیری امکان پذیر نباشد و برای وقتی هست که ما اطلاعات کمی از تابع ناشناخته داریم و در این حالت یادگیری غیر ممکن است و برای این مواقع ما فضای فرضیه را سعی میکنیم محدود کنیم به آن مواردی که میشناسیم تا یادگیری امکان پذیر شود.

پرسپترون اون بردار وزن را میچرخاند. آستانه یک هایپر پارامتر هست. دقت کن آستانه هم عین وزن ها یاد میگیریم که همان bias ما هست که در 1 ضرب میشود و میگیم از اون مقدار کمتر یا بیشتر باشد جدا میکنیم.

چرا آمار و احتمال؟ چون از یک سری داده میخواهیم یک قاعده کلی استخراج بکنیم. و تعمیم بدهیم به کل دیتا داشتیم . اون مواردی که از روش یاد گرفتیم داده آموزشی ما هستش.

جلسه دوم:

یادگیری زمانی استفاده میشود که 1. الگو وجود داشته باشد 2. نتوانیم به صورت ریاضی بنویسیم 3. داده برای اون الگو داریم. یادگیری نظارتی میگه قرار است ما ورودی را به خروجی نگاشت کنیم که این تابع f ما هست پس ما باید از فضای فرضیه یک تابعی به نام g را انتخاب کنیم که به اندازه کافی به تابع f نزدیک باشد. تابع ناشناخته همان f ما هست. learning میگوید حتما باید فضای فرضیه را کامل بگردی نه اینکه خود فرضیه را ثابت فرض کنی و به شما بدهند.

تابع g را learning پیدا کرده است. دقت کن که بر اساس اصولی که بهش رسیدیم اگر m خیلی بزرگ باشد یعنی اون فضای فرضیه ما اون عدد 1 میشود و یادگیری امکان پذیر نمیشود چون اون مقدار باید به صفر میل بکند تا ما خطا نداشته باشیم و حدسی که میزنیم با داخل کوزه یکسان باشد پس اگر تعداد n که داده های ما هست کم باشد به همان اندازه فضای m هم باید محدود در نظر بگیریم تا یادگیری ممکن شود.

در پرسترون فضای فرضیه نامحدود است و عملا این قاعده ممکن نیست.

جلسه سوم:

هر چه قدر فضای فرضیه بزرگتر نزدیک تر کردن Ein , Eout سخت میشود. Ein میشود ورودی و برداشتی من. Eout میشود اون تخمین واقعی از کل مسئله.

چرا Ein و Eout فاصله میگیرند گاهی اوقات؟ (نمودار رنگ صورتی و سبز منظورمه) چون داده جدید میبینیم نسبت به قبلی ها در ترین و امکان misclassify شدن هست در اون خطی که پرسپترون میکشد و تا اپدیت بکند در این نتیجه داده ها به خوبی جدایی پذیر خطی نیستند.

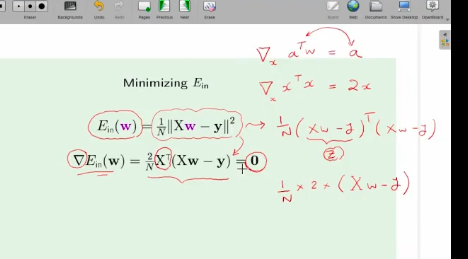
چرا پرسپترون خوب جدا نکرده؟ چون ممکن است در جای بدی متوقف شده باشد یا early stopping کرده باشد و تو جای بدی متوقف شده باشد و اگر چند تا iteration بیشتر میزد خطا کمتر بود.

تغییر در الگوریتم پرسپترون: بخاطر وجود این مشکل میایم یک تغییری ایجاد میکنیم، و میگیم کدوم یک از مدل ها کمترین Ein داشته و اون رو به عنوان جواب در نظر میگیریم. که به این الگوریتم میگوییم pocket که یعنی اون مدل بهتر را در جیب خودمان گذاشتیم و جلو میرویم و در نهایت از اون بهترین استفاده میکنیم.

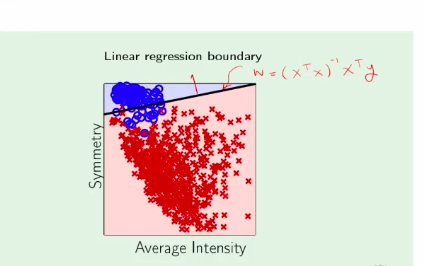
Regression: خروجی یک کلاس نیست یک مقدار حقیقی است. مثلا مقدار وامی که قرار است به آن شخص بدهیم. اون توان 2 علامت را خنثی میکند حالا چه h(x) بزرگتر از f(x) باشد یا نباشد. در این دست مسائل اگر Ein را کوچک بدست آوردی تقریبا میتوانی مطمئن شوی که Eout هم کوچک است. در دسته بند های خطی به جز وقتی که x خیلی بزرگ باشد بعد خیلی بزرگ باشد. ولی اگر کوچکتر از N باشد جای نگرانی نیست.

اگر ضرب داخلی اون بردار در وزن مثبت شد میشود جز کلاس 1 و اگر نشد میشود کلاس -1. X مثلا چون زاویه آن کمتر از 90 درجه هست میشود مال -1. اگر بیشتر از 90 درجه باشد 1. بردار وزن متعامد هست به زیر فضای خطی اون زیر فضای خطی ابر صفحه هست و مقدار ضرب داخلی آنها صفر میشود. تنها زیر فضای خطی که بعدش یکی کمتر از فضا هست میشود ابر صفحه.

نرم 2 به توان 2 تک به تک درایه ها را به توان 2 میرساند و با هم جمع میزد. که میشود همان 2 تا خط اطراف همون وزن منهای خروجی واقعی. برای کم کردن هم مشتق میگیریم.



رنک ماتریس میشود تعداد ستون هایی که از همدیگر مستقل خطی هستند. برای وقتی هست که تعداد داده ها خیلی بزرگتر از بعد هست و فیچر ها باید مستقل از هم باشند. فیچر ها نباید به همدیگه وابستگی خطی داشته باشند اگر این را تعریف کنیم میتوانیم بگوییم رنک ماتریس حداکثر d+1 شود که به معنی full rank بودن هست پس معکوس پذیر هست.



از regression برای کلاس بندی استفاده کردیم اما اشکال کجاست؟ چون در پایین قرمز ها مثلا خروجی تابع وزن در ورودی میشود -3 ولی این تابع میخواست -1 اتلاق کند به این نقاط و MSE زیاد میشود این باعث میشود دچار خطا شود و اذیت شود برای همین به یک جواب مطلوب نمیرسیم و معمولا از این برای وزن اولیه یا جواب اولیه برای الگوریتم پرسپترون استفاده میکنند.

مسائل در خطی با SINGLE SHOT حل میشوند یعنی کلا ورودی ضربدر وزن میکنی و دیگر iteration ندارد. مثلا برای مقدار ماندن در خانه طرف بعد 5 سال فرار نمیکند ولی این مقدارش با 20 سال در حالت خطی فرقی نمیکند یعنی اهمیتی ندارد 5 سال تا 20 سال ولی هر چه قدر بیشتر باشد این یعنی ماندگاری بالاتر است و برای ما مهم است پس میایم از مسائل غیر خطی استفاده میکنیم و مثلا میگیم طرف آیا کمتر از 1 سال و بیشتر از 5 سال ساکن بوده است؟

وقتی یک regression خطی هست نسبت به چی هست؟ نسبت به وزن ها یا ورودی؟ نسبت به وزن ها خطی هست دیگه چون داریم نسبت به وزن مشتق میگیریم. برای کلاس بندی هم همین است. حالا ورودی را میخواهیم ثابت فرض میکنیم و یک transformation غیر خطی بزنیم روی داده ها (اینم مشکلی نداره چون خطی بودن نسبت به وزن ها هست نه نسبت به ورودی ها مثلا فاصله داده از مراکز) و به فضای جدیدی برویم که و درفضای جدیدی جدای پذیر خطی هستند.

