

دانشگاه صنعتي امیرکبیر  
(پلی تکنیک تهران)

دانشكده مهندسی برق

ﮔﺰﺍﺭﺵکار تمرین عملی اول

مقدمه ای بر یادگیری ماشین

پیاده سازی Decision Tree

نگارش

علی بابالو

پویا ابراهیمی

استاد راهنما

استاد ساناز سیدین

آذر ماه 1401

در این تمرین قصد داریم تا الگوریتم درخت تصمیم را ساخته و آن را طبق داده ای که در اختیار داریم آموزش دهیم .

درخت تصمیم یکی از روش های یادگیری بانظارت است بدین معنی که داده های ما لیبل گذاری شده اند .

در این تمرین داده های ما دارای ۸ ویژگی هستند که به ۴ کلاس مختلف تقسیم بندی می شوند .

* **اضافه کردن کتابخانه**

برای ساخت درخت تصمیم از ۳ کتابخانه جامع پایتون استفاده می کنیم .

کتابخانه ی math برای انجام برخی محاسبات و توابع مهم ریاضیات

کتابخانه ی pandas برای خواندن دیتای مورد نیاز و تبدیل آن به دیتافریم

کتابخانه ی numpy برای استفاده از n-dim arrays برای سرعت و راحتی در برخی موارد

* **خواندن دیتاست**

مرحله بعد از اضافه کردن کتابخانه ها مرحله خواندن دیتاست است. در اینجا با لستفاده از تابع read\_csv() یک دیتافریم از داده ها ایجاد می­کنیم و آن را شافل و به دیتا های آموزش و تست دسته بندی می­کنیم. بشخصه از کتابخانه sklearn برای اینکار استفاده می­کنم اما بعلت شرط عدم استفاده از کتابخانه های دیگر با استفاده از تابع sample() دیتا را شافل و دسته بندی می­کنیم.

* Entropy & Information Gain

در این مرحله مقدار آنتروپی و Information Gain را محاسبه می­کنیم.

تابع Base\_entropy() مقدار آنتروپی کل تارگت دیتاست را محاسبه می­کند. ابتدا با یک حلقه تعداد مقادیر مختلف موجود در دیتا تارگت(خروجی) را محاسبه میکنیم سپس با استفاده از آن طبق رابطه زیر محاسبه میکنیم و آن را به عنوان خروجی تابع درنظر می­گیریم:

تابع entropy() وابسطه به تابع information\_Gain() است و در آن مقدار آنتروپی یک ویژگی با attribute مشخص را بدست می­آورد و آنتروپی آن ویژگی برابر با مجموع آنتروپی همه attribute های آن ویژگی است. در تابع entropy ابتدا تعداد attribute برای هر مقدار مشخص در ستون تارگت را بدست می­آوریم( مثلا برای feature = health و attribute = priority تعداد priority هایی که تارگت آن برابر not\_recom است را بدست می­آورد و اینکار را برای همه مقادیر یونیک در ستون تارگت بدست می­اورد و آن را در یک لیست ذخیره می­کند. در مرحله بعد مقادیر ۰ در لیستمان را حذف می­کنیم – ممکن است در یک attribute هیچ مقداری برای مقادیر موجود در تارگتمان نداشته باشیم- زیرا این ۰ در محاسبمان خطا ایجاد می­کند – در log گرفتن – بعد از آن بهمان روش تابع قبلی آنتروپی را حساب می­کنیم.

تابع information\_gain مقدار IG را حساب می­کند. همانطور که توضیح دادیم تابع entropy مقدار آنتروپی یک attribute خاص را محاسبه می­کند پس برای اینکه مقدار انتروپی کل آن feature را حساب کنیم نیاز به یک حلقه بر روی مقادیر یونیک آن feature داریم تا آنتروپی همه آن attribute ها را حساب و با هم جمع کنیم. سپس برای بدست آوردن IG، آنتروپی کل دیتاست را از انتروپی آن feature کم می­کنیم و آن را ریترن می­کنیم.

تابع find\_most\_informative\_feature() همانگونه که از اسمش مشخص است بهترین feature را ریترن می­کند. روش عملکردش هم به این صورت است که همه IG ها را بدست می­اورد و feature با بزرگترین بهره اطلاعاتی را برمی­گرداند.

* **ساخت درخت**

برای ساختن درخت از یک تابع بازگشتی استفاده می­کنیم. و این بازگشت را تا جایی که دیتاست خالی شود یا به ماکسیمم عمق درخت (تعداد feature) ها برسیم ادامه می­هیم. روش عملکرد آن هم بدین صورت است که ابتدا ویژگی با بهترین بهره اطلاعاتی را پیدا میکنیم سپس یک حلقه بر روی attribute های مختلف این ویژگی می­زنیم تا نود ها بدست بیایند و سپس در آن حلقه تابع ساخت درخت را دوباره صدا می­زنیم با این تفاوت که در دیتاست ورودی ویژگی که بهترین بهره اطلاعاتی داشته را حذف کردیم. (درخت را برای دیگر ویژگی ها ادامه میدهیم). بدین روش درخت ساخته می­شود