درخت تصمیم

استاد راهنما : دکتر وحید رضایی تبار

ارائه دهنده : حسام افشار

فهرست مطالب

1- مقدمه

2- الگوریتم کلی درخت تصمیم

3- ارزیابی یک درخت تصمیم

4- پروژه عملی

مقدمه

درخت تصمیم [1] (که **درخت پیش‌بینی** [2] نیز نامیده می‌شود) از ساختار درختی برای مشخص کردن توالی تصمیمات و پیامدها استفاده می‌کند. با توجه به ورودی $X=\{x_1,x_2, \dots , x_n\}$ هدف پیش بینی پاسخ یا متغیر خروجی Y است پس درخت تصمیم یک الگوریتم با نظارت است. پیش بینی را می توان با ساختن یک درخت تصمیم با نقاط آزمون و شاخه ها به دست آورد. در هر نقطه آزمایش، تصمیم گرفته می شود که شاخه خاصی را انتخاب کرده و درخت را طی کند. در نهایت به یک نقطه نهایی می رسد و پیشبینی انجام میشود. هر نقطه آزمون در درخت تصمیم شامل آزمون یک متغیر ورودی (یا ویژگی) خاص است و هر شاخه نشان دهنده تصمیمی است که گرفته شده است. مقادیر ورودی درخت تصمیم می تواند گسسته یا پیوسته باشد. به دلیل انعطاف پذیری و تجسم آسان، درخت های تصمیم معمولاً در برنامه های داده کاوی برای اهداف طبقه بندی استفاده میشوند.

1- Decision tree
2- Prediction tree

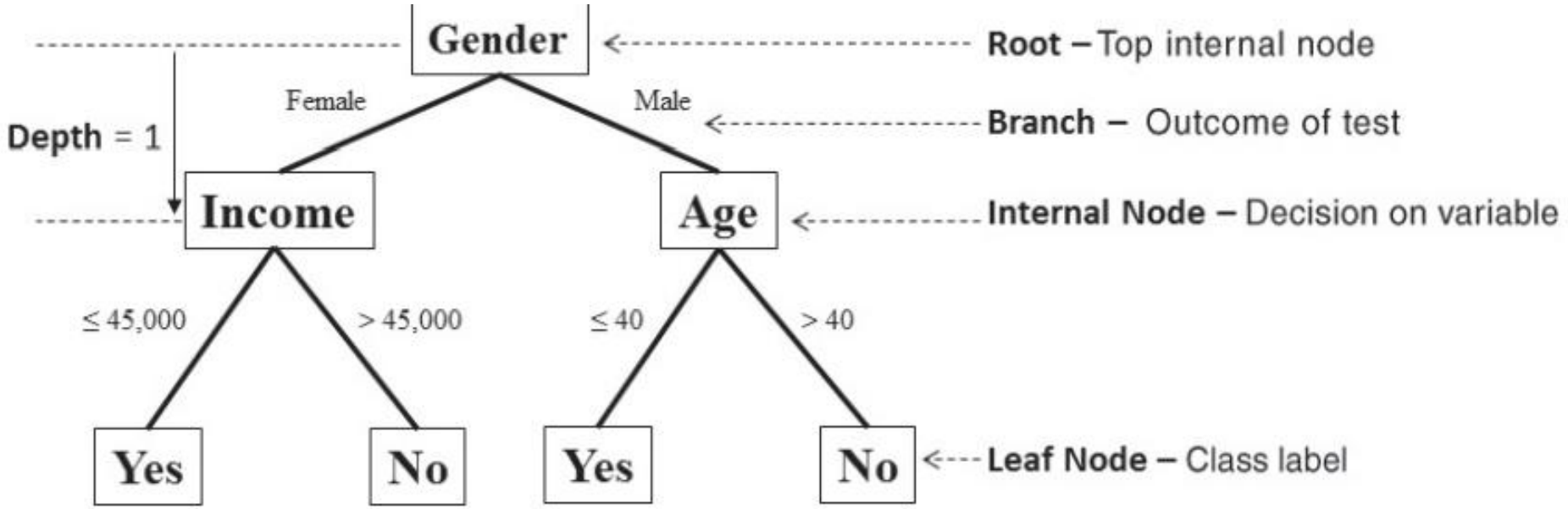
مقدمه

درخت تصمیم از ساختاری از نقاط تست (به نام **گره ها**[1]) و **شاخه ها**[2] استفاده می کند که نشان دهنده تصمیم گیری است. گره بدون شاخه را **گره برگ**[3] می نامند که برچسب های کلاس را برمی‌گردانند و در برخی از پیاده‌سازی‌ها، امتیازات احتمال را برمی‌گردانند. درخت های تصمیم دو گونه دارند: **درخت های طبقه بندی** و **درخت های رگرسیون**. درخت‌های طبقه‌بندی معمولاً برای متغیرهای خروجی اعمال می‌شوند که ماهیت دسته بندی دارند، مانند بله یا خیر، خرید یا عدم خرید، و غیره. از سوی دیگر، درخت‌های رگرسیون می‌توانند برای متغیرهای خروجی که عددی یا پیوسته هستند، مانند قیمت پیشبینی شده یک کالای مصرفی یا احتمال خرید اشتراک، اعمال شوند. علاوه بر این، از آنجا که نتیجه یک سری از گزاره های منطقی (اگر،آنگاه) است، هیچ فرض اساسی مبنی بر رابطه خطی (یا غیرخطی) بین متغیرهای ورودی و متغیر پاسخ وجود ندارد.

1- Nodes
2- Branch
3- Leaf node

مقدمه

شکل زیر نمونه ای از استفاده از درخت تصمیم را برای پیش بینی اینکه آیا مشتریان یک محصول را خریداری می کنند نشان می دهد. اصطلاح **شاخه** به نتیجه یک تصمیم اشاره دارد و به عنوان خطی که دو گره را به هم متصل می کند، تجسم می شود.



مقدمه

گره های داخلی[1] نقطه تصمیم گیری یا آزمون هستند. هر گره داخلی به یک متغیر ورودی یا یک ویژگی اشاره دارد. گره داخلی بالایی **ریشه**[2] نام دارد.

عمق یک گره حداقل تعداد مراحل لازم برای رسیدن به گره از ریشه است.

گره های برگ در انتهای آخرین شاخه های درخت قرار دارند. آنها نشان دهنده برچسب های کلاس هستند - نتیجه تمام تصمیمات قبلی. مسیر از ریشه تا گره برگ شامل مجموعه ای از تصمیمات گرفته شده در گره های داخلی مختلف است

1- Internal nodes
2- Root

الگوریتم کلی درخت تصمیم

به طور کلی، هدف یک الگوریتم درخت تصمیم، ساختن درخت T از مجموعه داده آموزش S است. اگر همه رکورد های S متعلق به کلاس C باشند یا اگر S به اندازه کافی خالص باشد (بیشتر از یک آستانه از پیش تعیین شده)، آنگاه آن گره یک گره برگ در نظر گرفته می شود و برچسب C به آن اختصاص می یابد.
خلوص[1] یک گره به عنوان احتمال آن از کلاس مربوطه تعریف می شود. در مقابل، اگر همه رکوردهای S متعلق به کلاس C نباشند یا اگر S به اندازه کافی خالص نباشد، الگوریتم آموزنده ترین ویژگی بعدی مثلا A را انتخاب میکند. و قسمت بندی طبق مقادیر A انجام میشود. الگوریتم زیردرخت های $T_1, T_2, \ldots$ را برای زیر مجموعه های S به صورت بازگشتی می سازد تا زمانی که یکی از شرایط توقف زیر رخ دهد:

تمام گره های برگ در درخت حداقل آستانه خلوص را برآورده کنند.

درخت را نتوان با حداقل آستانه خلوص از پیش تعیین شده بیشتر تقسیم کرد.

هر معیار توقف دیگری رعایت شود (مانند حداکثر عمق درخت).

1-Purity

الگوریتم کلی درخت تصمیم

اولین قدم در ساختن درخت تصمیم، انتخاب ویژگی با بیشترین اطلاعات مفید است. یک راه متداول برای شناسایی این ویژگی، استفاده از روش های مبتنی بر آنتروپی[1] است که توسط الگوریتم های یادگیری درخت تصمیم مانند ID3 و C4.5 استفاده می شود. روش های آنتروپی آموزنده ترین ویژگی را بر اساس دو معیار اساسی انتخاب می کنند:

- **آنتروپی**، که ناخالصی یک ویژگی را اندازه گیری می کند.
- **اطلاع سومندی[2]**، که خلوص یک ویژگی را اندازه گیری می کند

با توجه به کلاس Y و برچسب آن $y \in Y$ فرض کنید $P(y)$ احتمال y باشد. H_y آنتروپی y به صورت زیر تعریف میشود:

$$H_y = -\sum_{\forall y \in Y} P(y) \log_2 P(y)$$

1-Entropy
2-Information gain

الگوریتم کلی درخت تصمیم

مرحله بعدی شناسایی **آنتروپی شرطی** برای هر ویژگی است. با توجه به ویژگی X، مقدار آن x، متغیر خروجی آن Y و مقدار آن y آنتروپی شرطی $H_{Y|X}$ آنتروپی باقیمانده از Y با دادن X است که به صورت زیر نمایش داده میشود:

$$H_{Y|X} = \sum_{x} P(x)H(Y|X=x) = -\sum_{\forall x \in X} P(x) \sum_{\forall y \in Y} P(y|x) \log_2 P(y|x)$$

آنتروپی شرطی کوچکتر از آنتروپی پایه است. در بدترین حالت، وقتی ویژگی با نتیجه همبستگی ندارد، آنتروپی شرطی برابر با آنتروپی پایه است.

الگوریتم کلی درخت تصمیم

اطلاع سودمندی یک ویژگی A به عنوان تفاوت بین آنتروپی پایه و آنتروپی شرطی ویژگی تعریف می شود:

$$InfoGain_A = H_S - H_{S|A}$$

اطلاع سودمندی یک ویژگی عبارت است از مقدار کاهش آنتروپی که به واسطه جداسازی نمونه ها از طریق این ویژگی حاصل میشود. الگوریتم بر روی ویژگی با بیشترین میزان اطلاع سودمندی در هر دور تقسیم می شود. پس از هر تقسیم، الگوریتم به تمام رکوردها در یک گره برگ نگاه می کند و اطلاع سودمندی هر ویژگی کاندید دوباره بر روی این رکوردها محاسبه می شود. تقسیم بعدی بر روی ویژگی با بالاترین اطلاع سودمندی است. یک رکورد بعد از همه تقسیم‌ها فقط می‌تواند به یک گره برگ تعلق داشته باشد، اما بسته به پیاده‌سازی، یک ویژگی ممکن است در بیش از یک تقسیم درخت ظاهر شود. این فرآیند قسمت بندی رکوردها و یافتن آموزنده ترین ویژگی تکرار می شود تا زمانی که گره ها به اندازه کافی خالص باشند، یا با تقسیم بر ویژگی های بیشتر، اطلاعات کافی به دست نیاید. از طرف دیگر، می توان رشد درخت را زمانی متوقف کرد که تمام گره های یک گره برگ به یک کلاس خاص تعلق داشته باشند.

ارزیابی یک درخت تصمیم

چند راه برای ارزیابی درخت تصمیم وجود دارد. ابتدا ارزیابی کنید که آیا درخت منطقی است یا خیر.سپس به عمق و گره های درخت نگاه کنید. داشتن لایه های بیش از حد و به دست آوردن گره هایی با اعضای کم ممکن است نشانه های بیش برازشی باشد. در بیش برازشی، مدل به خوبی با مجموعه آموزشی مطابقت دارد، اما در نمونه‌های جدید در مجموعه تست عملکرد ضعیفی دارد. دو رویکرد می تواند به جلوگیری از بیش برازشی در یادگیری درخت تصمیم کمک کند:

- رشد درخت را زودتر قبل از اینکه به نقطه ای برسد که تمام داده های آموزشی کاملاً طبقه بندی شوند متوقف کرد.(مثلا گذاشتن شرط ماکسیمم عمق)
- درخت کامل را رشد دهید و سپس درخت را با روش هایی مانند **هرس با خطای کاهش یافته**[1] و **هرس با حداقل هزینه و پیچیدگی**[2] هرس کرد.

1- Reduced error pruning
2- Cost complexity pruning

ارزیابی یک درخت تصمیم

هرس با خطای کاهش یافته یکی از ساده ترین روش های هرس کردن است. با شروع از برگ ها، هر گره با محبوب ترین کلاس خود جایگزین می شود. اگر دقت پیش‌بینی تحت تأثیر قرار نگیرد، تغییر حفظ می‌شود. در حالی که هرس تا حدی ساده لوحانه است، اما هرس با خطای کاهش یافته مزیت سادگی و سرعت را دارد.

هرس با حداقل هزینه و پیچیدگی یک مجموعه از درخت های $T_0,T_1, \dots ,T_m$ را ایجاد می کند. که T_0 درخت اولیه و T_m فقط ریشه است. در مرحله i درخت با حذف یک زیر درخت از درخت مرحله i-1 و جایگزینی آن با محبوب ترین برگ ایجاد میشود. زیردرختی که حذف می شود به صورت زیر انتخاب می شود:

میزان خطای درخت T روی داده های S را به صورت $err(T,S)$ تعریف میکنیم.

زیر درخت t را به گونه ای انتخاب میکنیم که مقدار $\frac{err(prune(T,t),S)-err(T,S)}{|leaves(T)|-|leaves(pruen(T,t),S))|}$ را مینیمم کند.

که در آن $prune(T,t)$ درختی است که بعد از هرس کردن زیر درخت t از درخت T بدست می‌آید.

انتخاب اینکه درخت تا چه مرحله ای هرس شود را میتواند با اعتبار سنجی متقابل بدست آورد.

پروژه عملی

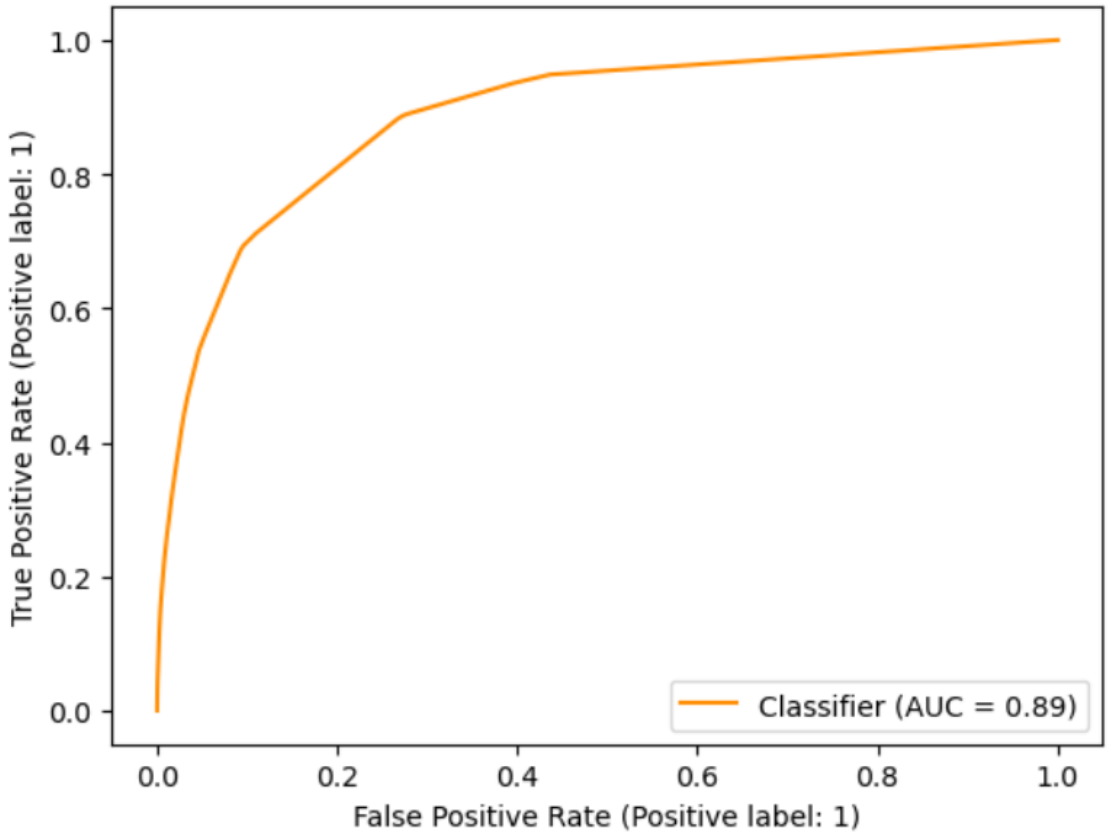
حال مدل درخت تصمیم را بر روی داده های بانک که داشتیم پیاده سازی میکنیم تا نتایج آن را بدست بیاوریم و با مدل های دیگر مقایسه کنیم. ابتدا با استفاده از داده های آموزش که در بخش رگرسیون لوژستیک تقسیم بندی کردیم بهترین پارامتر های مدل که همان ماکسیمم عمق و میزان هرس کردن درخت هستند را با استفاده از اعتبار سنجی متقابل میابیم.

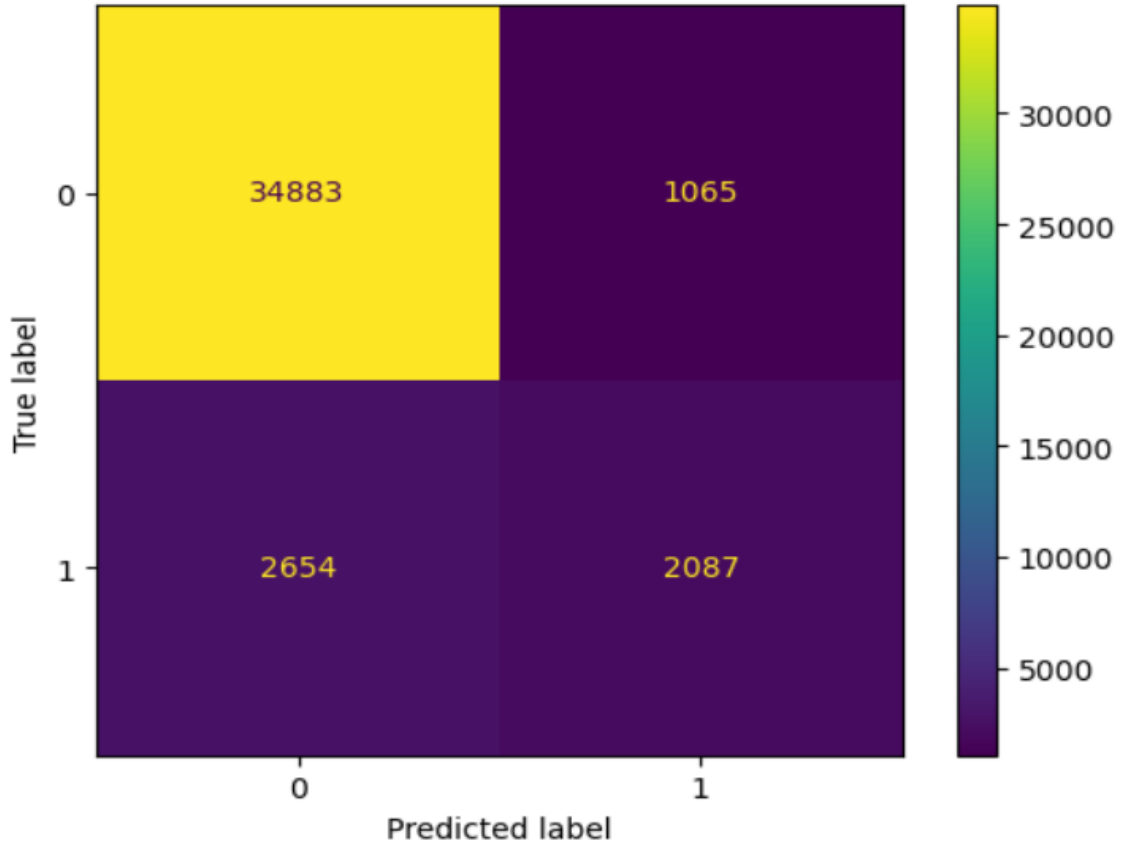
بهترین مدل بدست آمده درختی با ماکسیمم عمق 6 و آلفای هرس با حداقل هزینه و پیچیدگی 0 است.

که آلفای 0 به این معنی است که درخت نیاز به هرس شدن ندارد.

پروژه عملی

همچنین نمودار Roc و ماتریس درهم ریختگی برای داده های آموزش به صورت زیر بدست می‌آید:





پروژه عملی

و دقت بدست آمده برای داده های آموزش 91 درصد است که دقت بهتری نسبت به مدل رگرسیون لوژستیک است. و همچنین با نگاه به ماتریس درهم ریختگی مشاهده میشود که مدل به صورت تراز تری نسبت به مدل لوژستیک هر دو طبقه را پیشبینی کرده است.

همچنین برای اینکه مقایسه دقیق تری داسته باشیم مدل را با داده های آزمونی که به طور مشخص از داده های جدا کردیم و در برازش هیچ یک از مدل ها نقشی نداشته اند ارزیابی میکنیم. دقت بدست آمده برای داده های آزمون 89 درصد است که دقتی برابر با مدل لوژستیک و بهتر از مدل KNN دارد.

فایل داده ها و کد تحلیل های انجام شده با پایتون در لینک زیر قرار داده شده:

https://github.com/hesamafshar/Classification/tree/main/Banking%20Dataset%20-%20Marketing%20Targets