جواد فرجی 99522005

پروژه درخت تصمیم

مرحله اول) دیتاستی که داریم برای تحلیل و ساختن درخت مناسب نیست زیرا شامل مقادیر غیرعددی، عددهای بسیار بزرگ و تقریبا غیرقابل تحلیل، مقادیر پیوسته و هم چنین عناصر زائد است؛ بنابر این باید دیتاست را تمیز کنیم و مقادیر آن را به اعداد جدید و قابل فهم تری map کنیم تا تحلیل و طراحی درخت آسان تر شود. بدین منظور از فایل های onlinefraud_cleaning_dynamic.py و onlinefraud_cleaning_static.py می توان استفاده کرد. از فایل online_fraud_cleaning.ipynb برای آزمون و خطا استفاده شده و نتایج و کدهای بهینه در دو فایل مزبور آمده است. مقادیر nameorig و namedest به دلیل تنوع زیاد از دیتاست حذف شده اند، هر کدام از مقادیر type به یکی از اعداد 1 تا 5 map شده است. مقدار amount در 10 بازه دسته بندی شده و دو بازه هم کوچکتر از کمترین مقدار و بزرگتر از بیشترین مقدار در نظر گرفته شده است و هر یک از این بازه ها به یکی از مقادیر 0 تا 11 map شده است. بدین ترتیب 5 بازه هم برای oldbalanceorg ، oldbalancedest ، newbalanceorg ، newbalancedest در نظر گرفته شده است. این مقداردهی به صورت دستی در فایل onlinefraud_cleaning_static.py انجام شده است و چنان چه بخواهیم تعداد بازه ها را تغییر دهیم می توان از فایل onlinefraud_cleaning_dynamic.py استفاده کرد، در صورت زیاد کردن بازه ها به ویژه بازه های amount، ممکن است دقت درخت بالاتر رود. نتیجه این دسته بندی جدید در فایل های full_data_cleaned_dynamic.csv و full_data_cleaned_static.csv ذخیره شده تا بعدا مورد استفاده قرار گیرد.

مرحله دوم) با استفاده از معیار آنتروپی درخت خود را در فایل entropy.py می سازیم. از فایل entropy_and_giniindex.ipynb برای آزمون و خطا به خصوص در تعیین داده های train و test استفاده شده و نتایج در entropy.py آمده است. ابتدا فایل full_data_cleaned_static.csv را می خوانیم، سپس توابع محاسبه آنتروپی و دست آورد اطلاعات و هم چنین کلاس و تابع درخت تصمیم را تعریف می کنیم. درخت خروجی به صورت یک دیکشنری خواهد بود که keyهای آن مقادیر attributeها و Value های آن یا پاسخ نهایی (0 یا 1 به معنای جعلی بودن یا نبودن) و یا یک زیردرخت است. در ادامه تابع پیش بینی داده و هم چنین تابع محاسبه دقت داده های test تعریف شده اند.

ابتدا داده های درست و غلط را جدا کرده و 2400 داده تصادفی انتخاب می کنیم، نسبت داده های جعلی به کل داده ها 5 درصد است و می توان این مقادیر را تغییر داد

از روش stratified k-fold cross validation برای ساخت درخت استفاده می کنیم. در این روش دیتاست به k بخش مساوی (در اینجا 10) تقسیم می شود که در هر زیربخش هم نسبت داده های درست به غلط یکسان است. سپس k مرتبه عملیات ساخت درخت

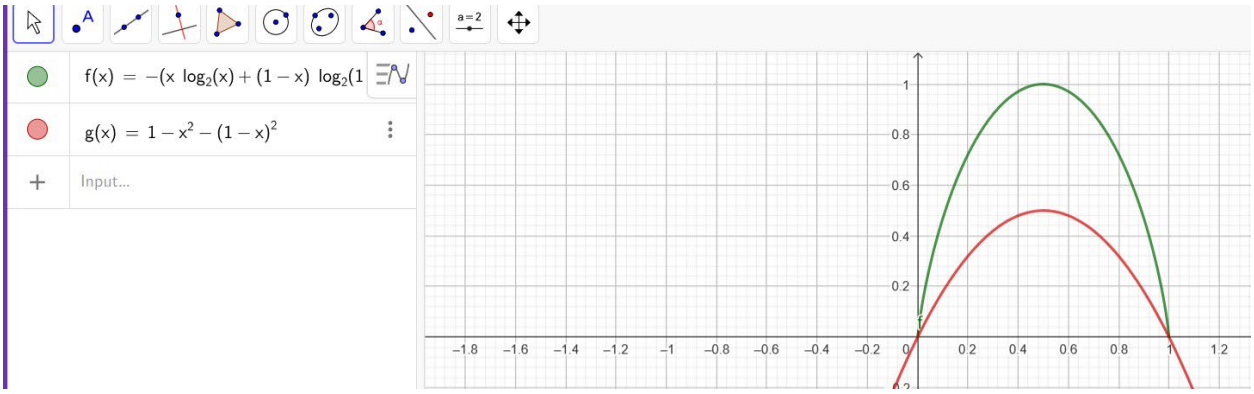
انجام می پذیرد که در هر مرحله یکی از زیربخش ها به عنوان داده test و بقیه زیربخشها به عنوان داده train استفاده می شود. در نهایت دقت درخت ها چاپ و مقایسه می شود و درختی که بیشترین دقت را داشته باشد چاپ می شود.

چون در داک پروژه تاکید شده بود که بهتر است در داده های test نسبت درستها به غلط ها یکی باشد، یک مجموعه داده test جدید 4000 تایی (این مقدار قابل تغییر است) از داده هایی که قبلا انتخاب نشده اند، بر می گزینیم و درختهای قبلی را تست می کنیم و دقتها را چاپ می کنیم. مشاهده می شود که مقادیر این دقتها از دقت های قبلی اندکی کمتر است زیرا که نسبت داده های غلط بالاتر رفته است. در نهایت درختی که بیشترین دقت را دارد چاپ می شود.

علت اینکه روش stratified k-fold را انتخاب کرده ایم این است که این روش در دیتاست هایی که نسبت داده های غلط به درست یا بالعکس خیلی کم است (imbalance data) کاربرد داشته و باعث ساخت درخت بهتری می شود. (در این دیتاست هم این نسبت حدود 2 درصد است)

مرحله سوم) همان کارهای مرحله قبلی منتهی به جای آنتروپی از gini index در gini_index.py استفاده شده است.

تفاوت آنتروپی و gini index : آنتروپی به دلیل اینکه یک تابع لگاریتمی است نسبت به gini index که یک تابع درجه 2 است، نسبت های کوچکتر را بیشتر برجسته می کند و برای دیتاست هایی که داده های نامتوازن دارند مثل دیتاست ما مناسب تر است. گرچه به دلیل لگاریتمی بودن اردر محاسباتی بالاتری دارد.



مشاهده می کنیم که آنتروپی مقادیر بیشتری نسبت به gini index در احتمالهای کوچکتر دارد.

مرحله چهارم) در نهایت باید درخت را رسم کنیم. در فایل visualize.py درخت به صورت یک گراف جهت دار رسم می شود. برگ ها جعلی بودن یا نبودن را مشخص می کنند و مستطیل هستند. سایر راسها attribute ها و مقادیر آنها را نشان می دهند و دایره هستند.

چالشها :

1) مهندسی داده ها و تمیز کردن درست آنها به شیوه ای که راحتتر قابل بررسی باشد، انتخاب درست تعداد بازه ها برای مقادیر عددی.

2) انتخاب درست داده های train و test : ابتدا از 2000 داده اول برای این کار استفاده شد اما مشاهده گردید که دقت درخت پایین است چون که داده ها اکثرا پراکندگی نداشته (به خصوص برای ویژگی step) و هم چنین نسبت داده های جعلی خیلی کم است. بنابر این به صورت دستی نسبت داده های جعلی بیشتر شد و هم چنین داده ها به صورت تصادفی انتخاب شدند.

3) رسم درخت : به صورتی که خواناتر باشد با تغییر شکل برگها به مستطیل و هم چنین انتخاب رنگ های متفاوت برای یالها