به نام خدا

تمرین دوم درس یادگیری ماشین (بخش اول گزارش)

نام : حسین سیم چی (۹۸۴۴۳۱۱۹)

نام استاد : آقای دکتر آبین

بهار ۱۳۹۹

مقدمه :

تمرین دوم در مورد پیاده سازی الگوریتم ID5 میباشد.

فایل گزارش همانند تمرین اول **شامل دو بخش** میباشد ، بخش اول که همین فایل میباشد و شامل گزارش مراحل انجام کار میباشد و بخش دوم نیز که در فرمت مقاله نوشته شده است شامل جزئیات مربوط به کد نوشته شده میباشد .

الگوریتم ID5 :

فرآیند کار این الگوریتم اینگونه میباشد که با ورود هر داده ی جدید الگوریتم فیچر یا ویژگی با کمترین آنتروپی را در ریشه قرار می دهد و سپس در مراحل بعدی نیز همین کار را تکرار می کند تا درخت به نود پایانی برسد . در ادامه با ورود هر داده ی جدید باید همین کار تکرار شود . علاوه بر این موضوع باید در هربار آپدیت فرزندان برای اینکه مطمئن باشیم بهترین فیچر را انتخاب می کنیم باید Information Gain مربوط به هر فیچر را محاسبه نموده و اگر Information Gain فرزندان از نود پدر بیشتر باشد باید آپدیت صورت گیرد و جای این نودها بایکدیگر عوض شود به همین دلیل برخلاف ID3 ، این الگوریتم افزایشی عمل می کند و به جای اینکه با دریافت کل داده ها به یکباره درخت را ترسیم کند به صورت افزایشی و با آپدیت درخت با هربار وارد شدن نمونه ی جدید اینکار را انجام می دهد .

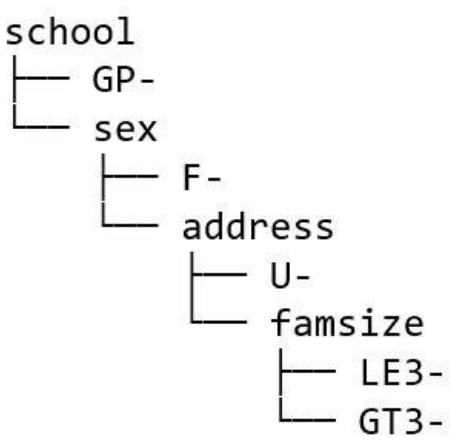
شرح کار انجام شده :

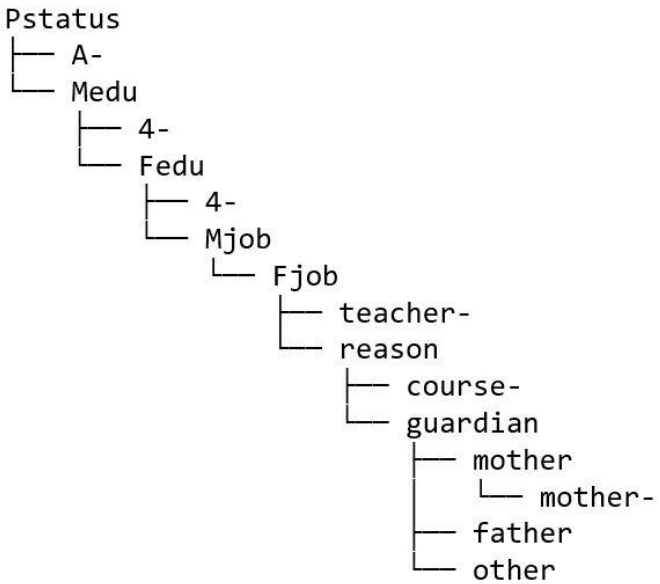
در ابتدا مطابق توضیحات گفته شده در بخش دوم گزارش ، با دریافت هر داده ، انتظار داریم فیچر با کمترین انتروپی یا بیشترین Information Gain بالاتر از بقیه فیچرها قرار گیرد و در مراحل بعدی و با دریافت داده ی جدید این درخت آپدیت شود تا در گام آخر با ورود آخرین داده درخت نهایی ترسیم شود و در اخر نیز با ورود داده ی تست ، دقت کار انجام شده را می یابیم .

با ورود اولین داده درخت ما مطابق شکل زیر ترسیم می شود :

```
  school sex address famsize Pstatus  Medu  Fedu     Mjob     Fjob  reason  \
0     GP   F       U     GT3       A     4     4  at_home  teacher  course

  guardian  Label
0   mother      0
```



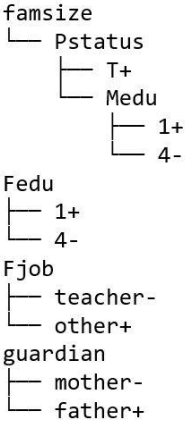


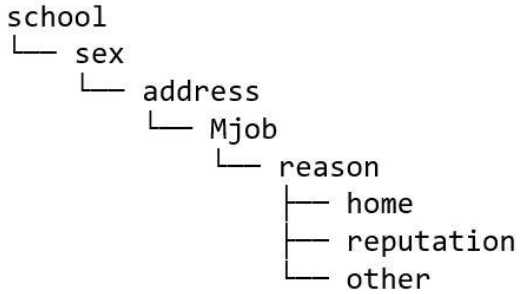
همانطور که از خروجی تصاویر مشخص است ، در اولین گام زمانی که داده ی اول وارد می شود چون تنها یک لیبل مشخص داریم به راحتی می توان با انواع مختلف از هر فیچر نتیجه را مشخص نمود به طوری که به عنوان مثال فیچر "school" که دارای بیشترین Information Gain میباشد در ریشه قرار می گیرد و بعد از آن فیچرهای دیگر قرار می گیرند . مثلا برای همین ویژگی یا فیچر قابل مشاهده است که با نوع "GP" میتوان لیبل آن را تشخیص داد . در ادامه با ورود هر داده ی جدید محل قرارگیری فیچرها آپدیت می شود و طبیعتا به دلیل افزایش داده ، پیچیدگی درخت ما افزایش می یابد .

با ورود داده ی جدید شکل درخت به صورت زیر آپدیت می شود :

	school	sex	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	reason	\
0	GP	F	U	GT3	A	4	4	at_home	teacher	course	
1	GP	F	U	GT3	T	1	1	at_home	other	course	

	guardian	Label
0	mother	0
1	father	1





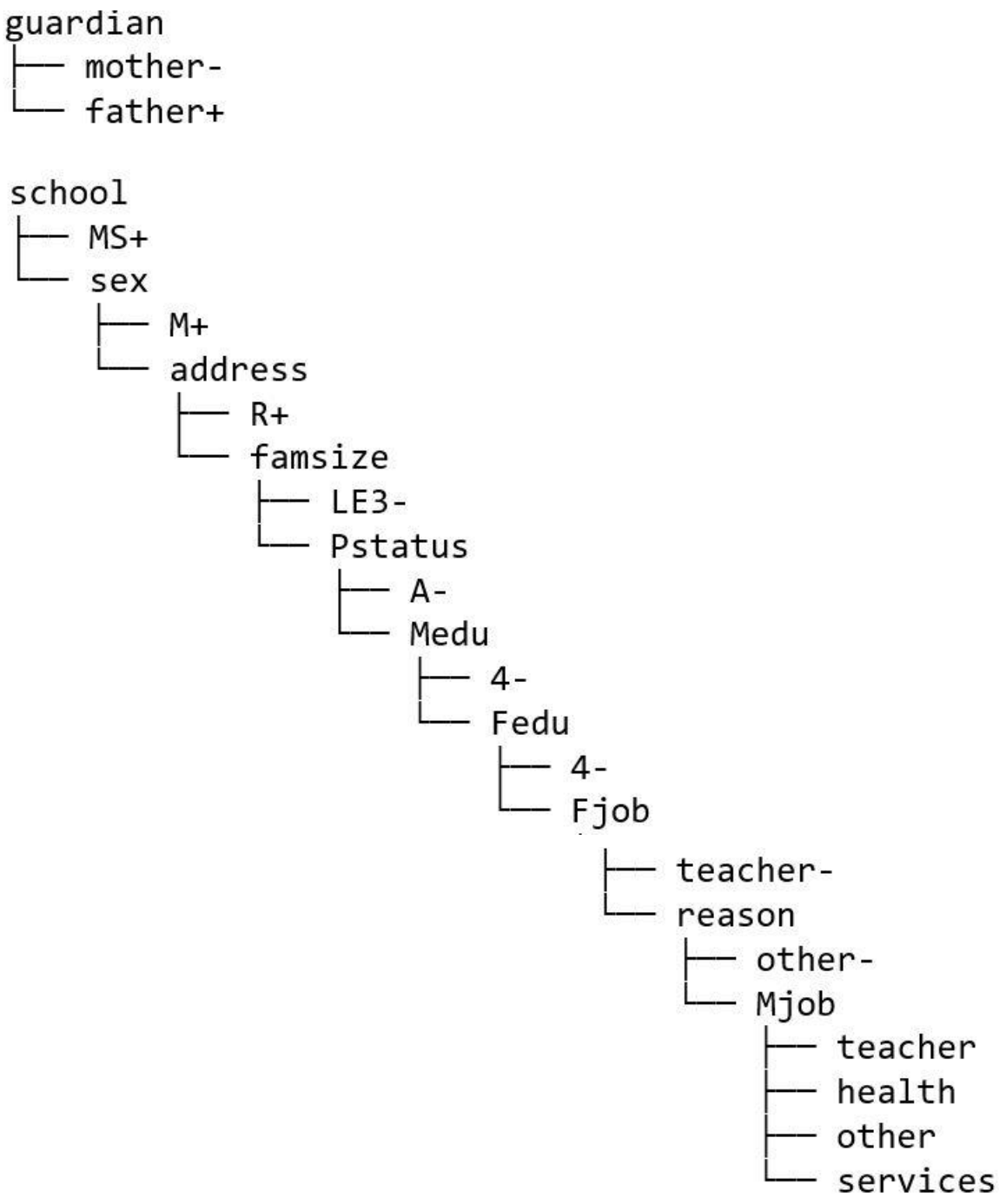
همانطور که از اولین و دومین شکل قابل برداشت است ، با ورود داده ی دوم شکل های درخت ما آپدیت شده است و محل قرار گیری فیچرها نیز بخاطر دلایل گفته شده تغییر کرده است .

ذکر این نکته ضروری است که در شکل بالا فیچر "school" در راستای فیچرهای دیگر قرار می گیرد و به دلیل اینکه میزان Information Gain آن از فیچر "guardian" کمتر است پایین تر از آن قرار می گیرد .

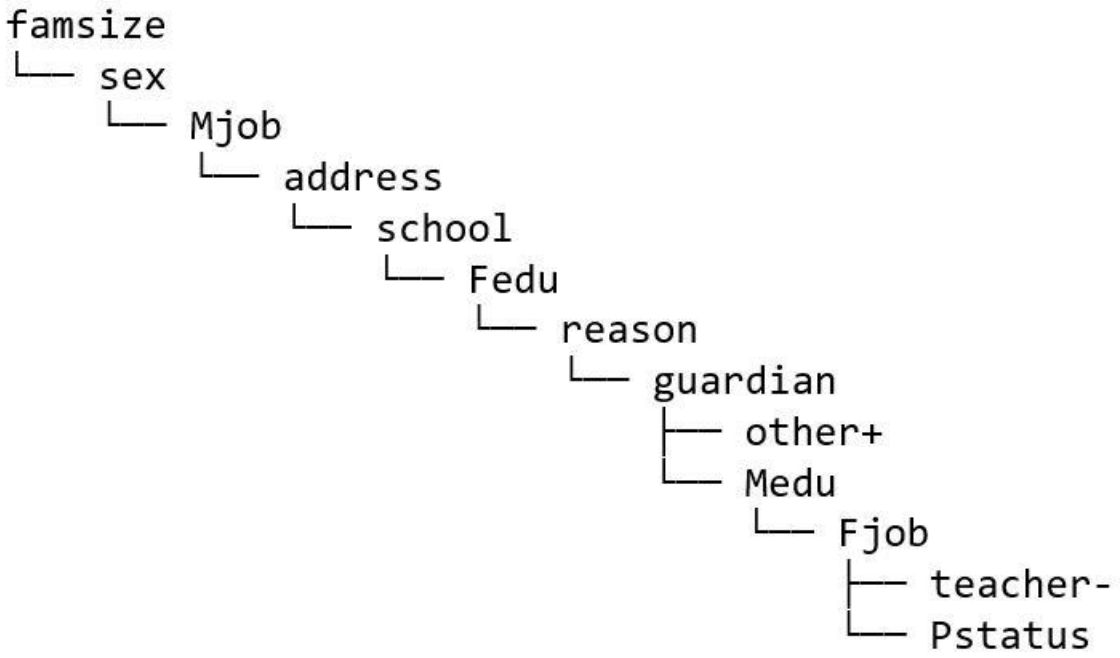
با ورود داده ی سوم ، درخت بازهم آپدیت میشود :

```
  school sex address famsize Pstatus  Medu  Fedu     Mjob     Fjob  reason  \
0     GP   F       U     GT3       A     4     4  at_home  teacher  course
1     GP   F       U     GT3       T     1     1  at_home    other  course
2     GP   F       U     LE3       T     1     1  at_home    other   other

  guardian  Label
0   mother      0
1   father      1
2   mother      0
```


guardian
mother-
father+
school
MS+
sex
M+
address
R+
famsize
LE3-
Pstatus
A-
Medu
4-
Fedu
4-
Fjob
teacher-
reason
other-
Mjob
teacher
health
other
services

این مراحل تا اخرین داده ی موجود تکرار می شود و در مرحله اخر با ورود اخرین داده ی موجود در مرحله آموزش شکل به صورت زیر تکمیل و آپدیت می شود :



ممکن است سوال اصلی این باشد که چرا از فیچر اول به دوم تنها یک مسیر داریم ، پاسخ اینگونه میباشد که فیچر اول نمیتواند با هر دو ویژگی خود درخت را به لیبل مثبت یا منفی تقسیم کند ، درنتیجه به سراغ فیچر های بعدی با عمق های بیشتر میرود و مفهوم در واقع به این شکل است که میتوان با هر دو ویژگی از فیچر اول به سمت فیچر دوم رفت و فرقی نمی کند که با کدام ویژگی اینکار صورت گیرد .

برای بدست اوردن دقت راهکارهای متفاوتی وجود دارد ولی همانطور که در بخش دوم گزارش نیز ذکر شده است در اینجا ما از فرمول استاندارد موجود precision برای بدست اوردن دقت استفاده کرده ایم .

با درنظرگرفتن بازه های مختلف برای داده های تست ، و همینطور استفاده از تکنیک cross validation و با درنظر گرفتن k های گوناگون دقت های متفاوتی بین بازه ی ۵۵ تا ۶۷ بدست می آوریم که چند مورد آن در زیر آمده است :

```
The prcision on test data is 0.54347826086956 52
The prcision on test data is 0.631578947368421
The prcision on test data is 0.64
The prcision on test data is 0.6666666666666666
```

در پایان از تلاش های تمامی دوستان سپاسگزارم.