دانشگاه صنعتی امیرکبیر

دانشکده مهندسی کامپیوتر

گزارش تکلیف سوم درس یادگیری ماشین

آشنائی با دسته‌بندی‌کننده‌های TAN و NB

دانشجو:

سید احمد نقوی نوزاد

ش-د: ۹۴۱۳۱۰۶۰

استاد:

دکتر ناظرفرد

بهار ۹۵

نکته: تمامی کدهای اصلی پروژه جدای از زیرتوابع نوشته‌شده در فایل 'main1.m' قرار گرفته‌اند.

سوال اول

الف) محاسبه‌ی مقدار CMI(Conditional Mutual Information)

مقدار **CMI** بین هر جفت از ویژگی‌ها با استفاده از معادله‌ی زیر محاسبه می‌شود و جدول مربوطه نیز به دنبال آن می‌آید:

$$CMI(X,Y|C) = \sum_{x,y,c} p(x,y,c) \log_2 \frac{p(x,y|c)}{p(x|c)p(y|c)}$$

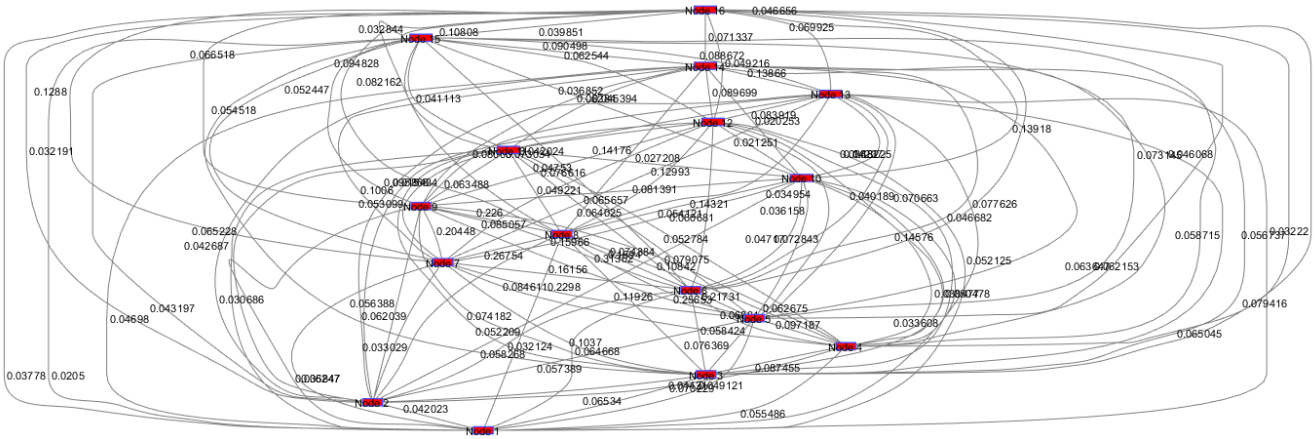
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1		0.04	0.07	0.05	0.04	0.06	0.05	0.05	0.03	0.03	0.04	0.05	0.03	0.04	0.02	0.03
2			0.04	0.06	0.05	0.05	0.06	0.03	0.05	0.03	0.06	0.01	0.09	0.03	0.04	0.03
3				0.08	0.07	0.05	0.10	0.11	0.07	0.04	0.03	0.07	0.05	0.06	0.05	0.07
4					0.09	0.06	0.08	0.07	0.06	0.03	0.05	0.05	0.07	0.06	0.04	0.05
5						0.21	0.22	0.31	0.25	0.03	0.04	0.07	0.13	0.13	0.07	0.06
6							0.16	0.15	0.15	0.04	0.06	0.10	0.14	0.14	0.07	0.04
7								0.26	0.20	0.06	0.05	0.08	0.12	0.10	0.09	0.12
8									0.22	0.05	0.04	0.08	0.14	0.14	0.08	0.10
9										0.06	0.06	0.07	0.08	0.06	0.05	0.06
10											0.02	0.02	0.03	0.02	0.03	0.04
11												0.07	0.04	0.04	0.04	0.03
12													0.08	0.08	0.06	0.04
13														0.13	0.08	0.06
14															0.09	0.07
15																0.03
16																

جدول شماره ۱: مقادیر **CMI** برای هر جفت از ویژگی‌های مجموعه‌ی داده

با توجه به متقارن‌بودن ماتریس مربوطه و نیز بی‌معنا بودن **CMI** برای یک ویژگی منحصر به فرد از ذکر درایه‌های قطر اصلی و مثلث پایینی ماتریس حاصله خودداری کردیم.

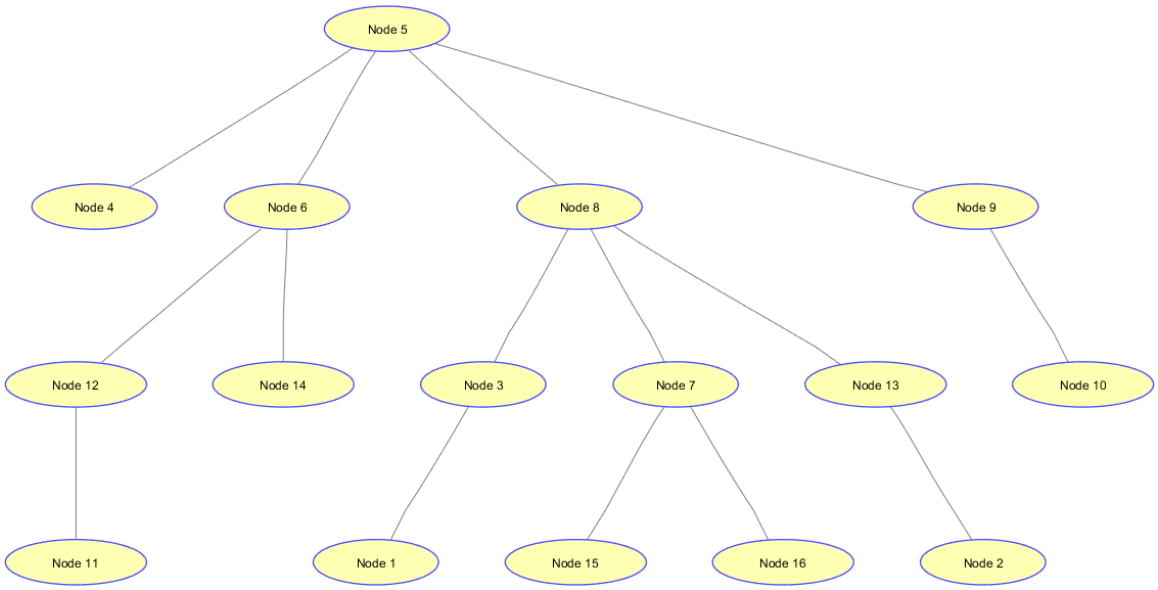
ب) گراف کامل متشکل از نودهای متناظر با هر کدام از ویژگی‌ها

در این‌جا گراف کاملی را تشکیل می‌دهیم که نودهای آن متناظر با هر کدام از ویژگی‌ها بوده و وزن هر کدام از یال‌های آن برابر مقدار **CMI** بین دو رأس یال مربوطه می‌باشد و قصد داریم از روی آن درخت پوشای با وزن بیشینه را به دست آوریم.



ج) اعمال الگوریتم Maximum Weighted Spanning Tree بر روی گراف حاصل از مرحله‌ی قبل

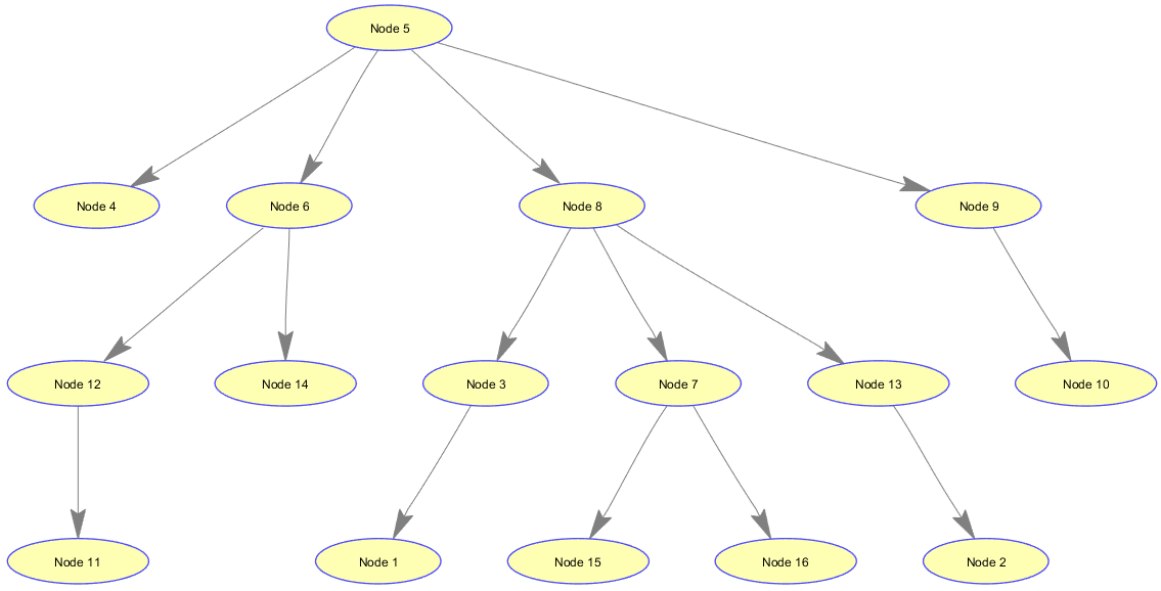
با اعمال الگوریتم **Maximum Weighted Spanning Tree** بر روی گراف کامل حاصل از مرحله‌ی پیشین، درخت بدون جهت حاصله به صورت زیر خواهد بود که در نهایت ما را به ساختار **TAN** رهنمون خواهد شد:



د) تبدیل درخت حاصل از مرحله‌ی قبل به یک درخت جهت‌دار

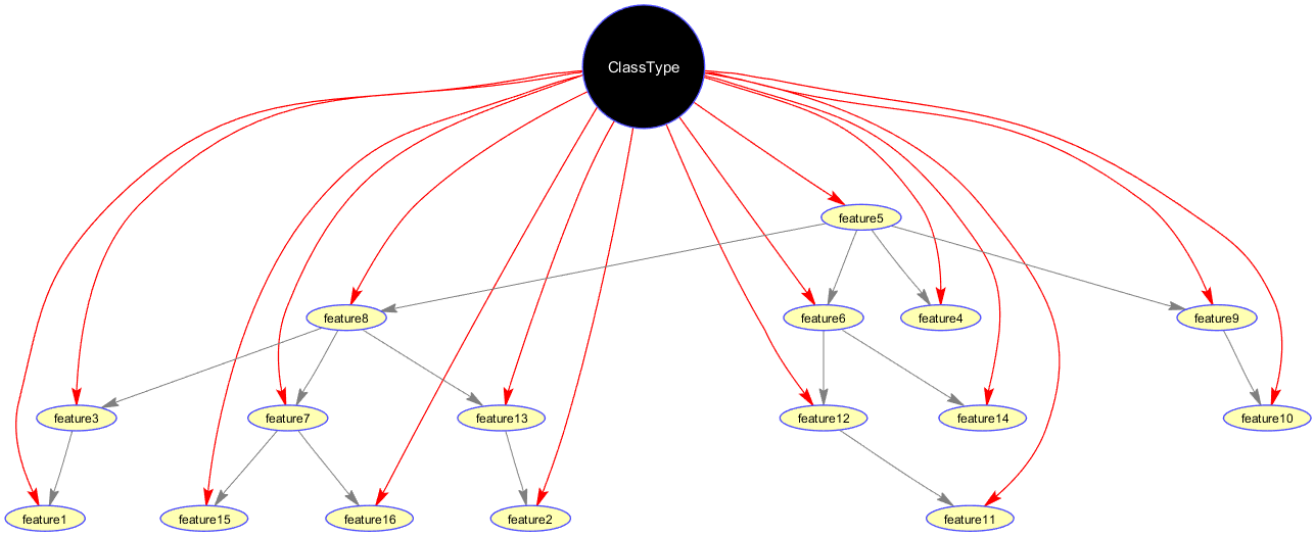
در اینجا برای تبدیل درخت بی‌جهت حاصل از مرحله‌ی پیشین به یک درخت جهت‌دار می‌توانیم یکی از گره‌ها را به صورت تصادفی به عنوان ریشه برگزیده و جهت یال‌های آن را تا آخرین سطح به صورت خروجی قرار دهیم که ما در اینجا گره‌ای را که بیشترین درجه را داراست به عنوان ریشه انتخاب کردیم.

لازم به ذکر است که ما در اینجا برای یافتن درخت مربوطه از دستور **biograph** نرم‌افزار متلب بهره بردیم که ورودی آن یک ماتریس مجاورت (در اینجا همان ماتریس **CMI**) بوده و خروجی آن گراف مرتبط با ماتریس ورودی می‌باشد. از جمله زیرتوابع دستور **biograph** ، تابعی با نام **minspantree()** می‌باشد که درخت پوشای با وزن کمینه را به دست می‌دهد و ما برای اینکه به درخت پوشای با وزن بیشینه دست یابیم، حیلتی خاص به کار بردیم و آن این بود که ابتدا تمامی وزن‌ها را که در بازه‌ی **(a,b) | a,b>0** بودند قرینه نموده و سپس با دو برابر مقدار بیشینه‌ی وزن‌ها یعنی با **2b** جمع نمودیم، تا در نهایت تمامی وزن‌ها در بازه‌ی جدید **(b,2b-a)** قرار گیرند و بدین‌گونه جای تمامی وزن‌های بیشینه و کمینه عوض شده و به عبارتی ترتیب چینش وزن‌ها معکوس گردید. در نهایت زیرتابع نامبرده یعنی **minspantree()** را بر روی گراف نهائی با وزن‌های تغییریافته اجرا نمودیم و درخت پوشای با وزن بیشینه حاصل گردید. اما از آنجا که برای استفاده از تولباکس **Bayes** نیاز به این درخت داشتیم، می‌بایست وزن‌ها را دوباره به حالت اولیه بازمی‌گرداندیم، که برای این کار دوباره همان عملیات مذکور یعنی قرینه‌سازی و جمع با دوبرابر بیشینه‌ی وزن‌ها یعنی **2b** را انجام دادیم و درخت مربوطه با وزن‌های صحیح حاصل گردید. درخت نهائی به صورت زیر می‌باشد:

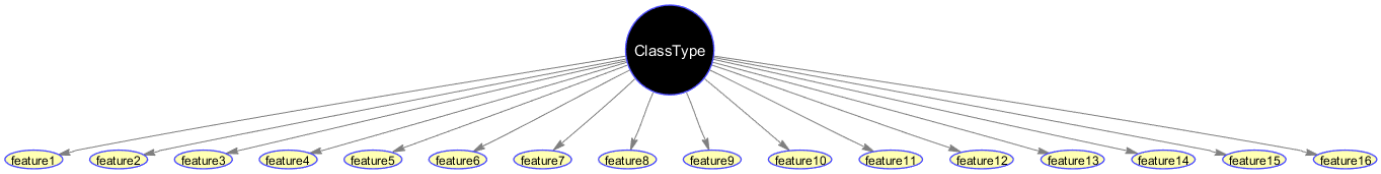


ه) تشکیل ساختار نهائی TAN(Tree Augmented Naïve Bayesian)

در اینجا برای تشکیل ساختار نهائی **TAN** تنها کافی است که یک گره‌ی اضافی متناظر با کلاس داده‌های آموزشی را به درخت جهت‌دار مرحله‌ی قبل اضافه کرده و از این گره به سایر گره‌ها یالی را رسم نمائیم که درنهایت درخت زیر حاصل می‌گردد:



لازم به ذکر است که اگر یال‌های مابین سایر گره‌ها متناظر با ویژگی‌ها را حذف نمائیم، در آن‌صورت ساختار درخت حاصله، همان ساختار **Naïve Bayesian** می‌باشد که در آن وابستگی بین ویژگی‌ها را نادیده گرفته‌ایم و نمودار آن نیز به صورت زیر می‌باشد:



قسمت دوم:

در اینجا برای دسته‌بندی یک داده‌ی آزمایشی به روش **leave-one-out** ، به تعداد دفعات برابر با تعداد داده‌های موجود در مجموعه داده مورد استفاده، هر بار یک داده را به عنوان داده‌ی آزمایشی بیرون کشیده و مابقی داده‌ها را به عنوان داده‌های آموزشی مورد استفاده قرار می‌دهیم؛ بدین گونه که با استفاده از مجموعه داده‌های آموزشی و نیز زیرتوابع کاربردی تولباکس مورد استفاده در این پروژه، جداول احتمالات شرطی مورد نیاز برای استنتاج در مورد داده‌ی آموزشی را یاد گرفته و در نهایت برای تعیین کلاس داده‌ی آزمایشی مورد استفاده تصمیم‌گیری می‌نمائیم. که البته احتمال داده‌ی تست مربوطه با استفاده از فرمول زیر محاسبه می‌گردد:

$$k = argmax\, p(C_k|X_1, X_2, \dots, X_n) = argmax \prod_{i=1}^{n} p(X_i|pa(X_i), C_k).p\big(pa(X_i), C_k\big)$$

$$(where \;\; pa(X_i) \;\; means\; parent\; node\; of\; node\; X_i)$$

در نهایت به ذکر نتایج نهائی برای هر دوی روش‌های دسته‌بندی **TAN**و **NB** بسنده می‌نمائیم:

	TAN Structure	NB Structure
Correctly Classified	400	393
Incorrectly Classified	35	42
Accuracy	91.95	90.34

همانطور که مشاهده می‌گردد دسته‌بند تکامل‌یافته‌ی **TAN** نسبت به نوع ساده‌تر آن یعنی **NB** با اندکی تفاوت بهتر عمل نموده است که البته این نشان از آن دارد که عملیات انجام‌شده در این پروژه خوشبختانه صحت داشته و البته حجم مجموعه داده‌ی مورد استفاده نیز مناسب حال بوده است، چرا که بنا به مشاهدات در برخی موارد که حجم مجموعه داده‌ی مورد استفاده اندک بوده است، دسته‌بند **NB** نتایج بهتری را نسبت به نوع تکامل‌یافته یعنی **TAN** حاصل نموده است.

!!! توجه !!!

در این‌جا می‌توانستیم روش leave-one-out را به صورت دیگری نیز پیاده‌سازی نمائیم؛ یعنی در هر مرحله از جداسازی مجموعه داده‌ی آموزشی و یکتاداده‌ی آزمایشی، ابتدا با استفاده از مجموعه‌ی داده‌ی آموزشی جدید، درخت مربوطه را ساخته و سپس عملیات مربوطه را با استفاده از تولباکس مربوطه ادامه می‌دادیم، و به عبارتی به تعداد داده‌های موجود در مجموعه داده‌ی vote (در این‌جا ۴۳۵ داده)، درخت‌های احتمالا متفاوت به دست می‌آوردیم.

لذا در این‌جا روش دوم گفته‌شده را نیز پیاده‌سازی نموده و در فایلی با نام 'main2.m' قرار داده‌ایم.

نتایج نهائی حاصل از این روش با روش اول چندان تفاوتی نداشتند.