Amirkabir University of Technology
(Tehran Polytechnic)

Department of
Computer Engineering

Course : Machine Learning

Homework 2

Najmeh Mohammadbagheri

99131009

بخش تشریحی

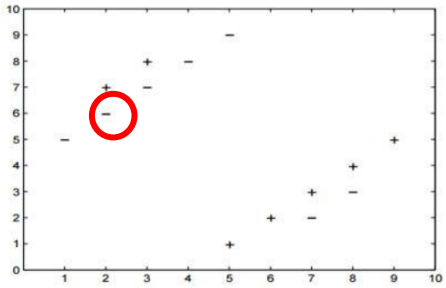
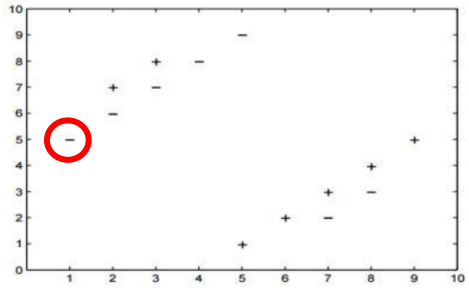
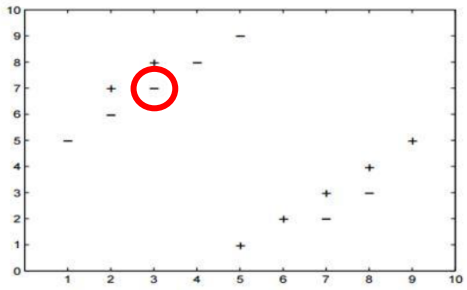
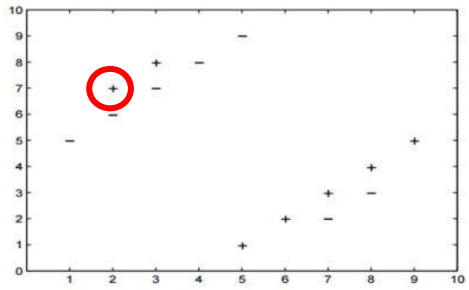
سوال اول

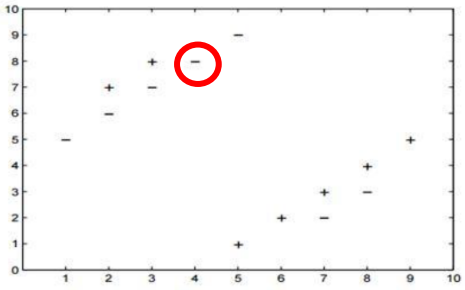
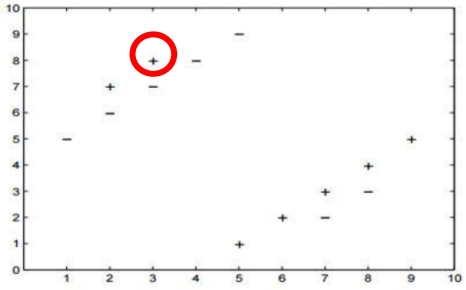
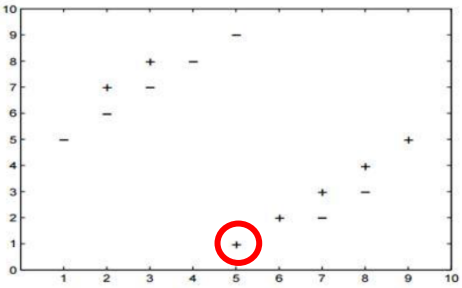
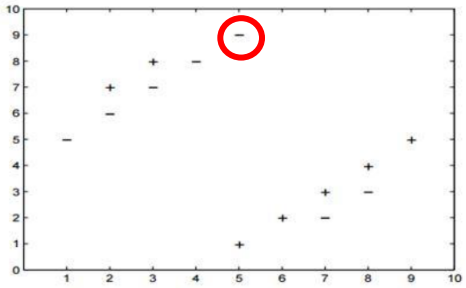
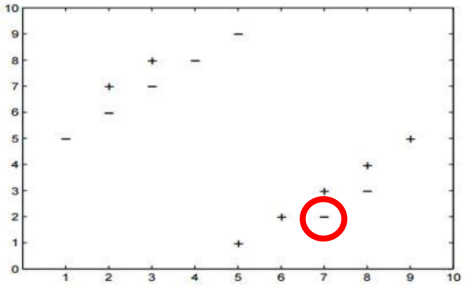
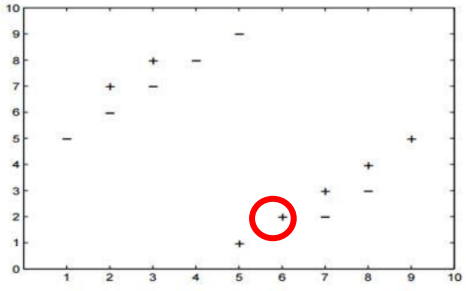
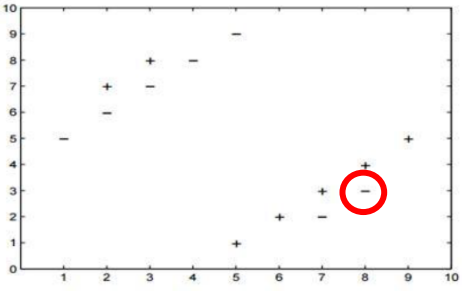
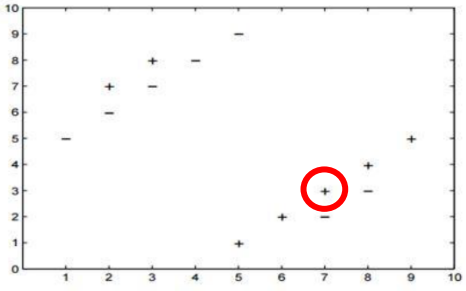
استفاده از الگوریتم Leave One Out Cross Validation.. در این روش اگر مجموعه داده‌ی ما n نمونه داشته باشد، به ازای هر نمونه یک بار فرایند یادگیری را تکرار می‌کنیم تا در نهایت بهترین پارامترها را انتخاب کنیم. در هر تکرار یک داده به عنوان داده‌ی تست جدا می‌شود و n-1 داده‌ی باقی مانده به عنوان داده‌ی آموزش استفاده می‌شود. برای این مثال خاص، در هر یادگیری مقادیر مختلف k را در نظر می‌گیریم. در آخر k مربوط به بهترین دقت را به عنوان جواب نهایی اعلام می‌کنیم. این روش بهینه‌ترین جواب را میدهد.

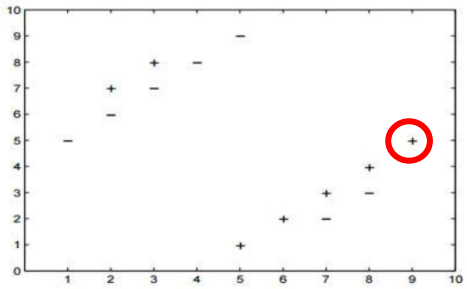
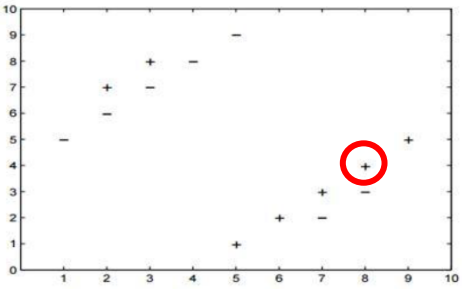
سوال دوم

الف. برای حل این سوال به ازای هر داده یکبار بهترین k را اعلام میکنیم. در هر تصویر داده‌ی مشخص شده با دایره قرمز، داده‌ی تست است و k مشخص شده مقداری است که تشخیص درست انجام شود.

جدول ۱ استفاده از الگوریتم Loocv

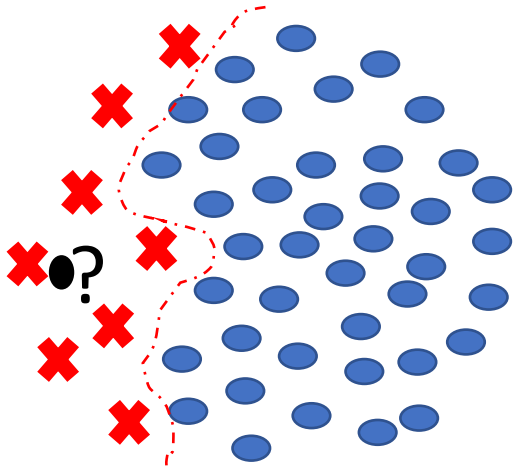
k = 3,5 	k = 1,3,5 
k = 5,6 	kای وجود ندارد که این نمونه صحیح برچسب بخورد. 

k = 3, 5 	k ای وجود ندارد که این نمونه صحیح برچسب بخورد. 
k = 1,3,5 	k = 1,3,5 
k ای وجود ندارد که این نمونه صحیح برچسب بخورد. 	k = 1,3,5 
k ای وجود ندارد که این نمونه صحیح برچسب بخورد. 	k = 5 

k = 1,3,5	k = 3,5
	

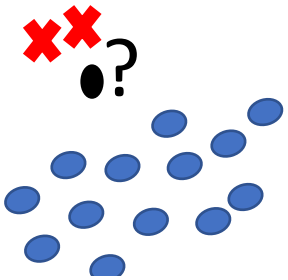
همانطور که در ۱۴ نمونه‌ی بالا مشاهده می‌کنیم به جز مواردی که k مناسب ندارند، در دیگر نمونه‌ها k=5 وجود دارد بنابراین با انتخاب k=5 بهترین برچسب‌دهی انجام می‌شود. چون در ۱۰ مورد تخمین درست انجام می‌شود بنابراین دقت حاصل برابر با ۱۰/۱۴ است.

ب. به ازای انتخاب kهای بزرگ مرزهای تصمیم smooth می‌شوند و این باعث می‌شود که انتخاب کلاس خطای بالایی داشته باشد. مثلا در تصویر ۱ اگر k برابر با ۲۰ انتخاب شود، تصمیم گیری برای نقطه‌ی a اشتباه انجام می‌شود. در واقع در نمونه‌ی زیر داده‌ی جدید باید برچسب کلاس قرمز را بگیرد ولی چون چگالی کلاس قرمز کمتر از آبی است، با انتخاب ۲۰ نزدیک‌ترین همسایه داده‌ی جدید برچسب آبی میخورد. همچنین ۲۰ از دو برابر تعداد اعضای کلاس قرمز نیز بیشتر است. پس در این حالت ها انتخاب عدد بزرگ برای k مناسب نیست.



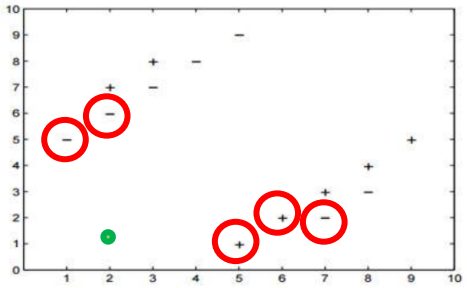
شکل ۱تاثیر انتخاب k بزرگ در تصمیم گیری

از طرفی نیز انتخاب k کوچک باعث حساس شدن تصمیم به داده‌ی نویز می‌شود. مثلا در شکل ۲ داده‌ی جدید برچسب آبی باید بگیرد ولی اگر k یک یا دو انتخاب شود، داده‌های نویز به عنوان همسایه در نظر گرفته می‌شوند و داده‌ی جدید برچسب قرمز می‌گیرد.



شکل ۲ تاثیر k کوچک بر تصمیم گیری

ج. باتوجه به شکل ۳ تعداد منفی ها بیشتر است، پس برچسب منفی به این داده زده می‌شود.



شکل ۳ انتخاب برچسب برای داده جدید با k = 5

سوال سوم

هردو غیرپارامتریک هستند ولی هر دو هایپرپارامتر دارند که در درخت تصمیم، عمق درخت و در K نزدیک‌ترین همسایه، K است. به این دو مورد از این جهت هایپر پارامتر گفته میشود که میزان واریانس و بایاس با تغییر آنها، تغییر می‌کند، و از این جهت غیرپارامتریک هستند که مجهولی برای یادگیری ندارند.

سوال چهارم

مدل‌های مولد توزیع واقعی داده‌ها را مدل می‌کنند و مدل‌ها تمایزگر یک مرز تصمیم را مدل می‌کنند. بنابراین هر دو الگوریتم متمایزگر هستند.

سوال پنجم

بله تنبل است. به الگوریتم‌هایی که از قبل یادگیری را روی داده‌ها انجام نمی‌دهند و زمان وارد شدن داده‌ی تست یا همان داده‌ی جدید عملیات خود را شروع می‌کنند. در این الگوریتم زمانی که داده‌ی جدید وارد می‌شود فاصله‌ی آن با تمام داده‌ها محاسبه می‌شود و سپس تصمیم گیری انجام می‌شود. به همین علت می‌گوییم تنبل است.

سوال ششم

از بیش برازش شدن جلوگیری می‌کند. زمانی که عمق درخت زیاد می‌شود لازم است هرس کردن انجام شود. با هرس کردن شاخه‌هایی که داده‌های کمی دارند و به یک پدر متعلق هستند را یکی میکند و برای هر کلاس احتمال آنرا نشان می‌دهد.

سوال هفتم

$$E(s) = -\frac{5}{14}\log\left(\frac{5}{14}\right) - \frac{9}{14}\log\left(\frac{9}{14}\right) = 0.925$$

Gain calculation:

Age

$$E_{youth} = -\left(\log\left(\frac{3}{5}\right) * \left(\frac{3}{5}\right) + \log\left(\frac{2}{5}\right) * \left(\frac{2}{5}\right)\right) = 0.970$$

$$E_{senior} = -\left(\log\left(\frac{3}{5}\right) * \left(\frac{3}{5}\right) + \log\left(\frac{2}{5}\right) * \left(\frac{2}{5}\right)\right) = 0.970$$

$$E_{youth} = 0$$

$$E_{Age} = \frac{4}{14} * 0 + \frac{5}{14} * 0.97 + \frac{5}{14} * 0.97 = 0.7$$

$$\text{Gain(Age)} = 0.925 - 0.7 = 0.225$$

income

$$E_{high} = -\left(\log\left(\frac{2}{4}\right) * \left(\frac{2}{4}\right) * 2\right) = 1$$

$$E_{medium} = -\left(\log\left(\frac{4}{6}\right) * \left(\frac{4}{6}\right) + \log\left(\frac{2}{6}\right) * \left(\frac{2}{6}\right)\right) = 0.9$$

$$E_{low} = (-\left(\log\left(\frac{3}{4}\right) * \left(\frac{3}{4}\right) + \log\left(\frac{1}{4}\right) * \left(\frac{1}{4}\right)\right)) = 0.815$$

$$E_{income} = \frac{4}{14} * 1 + \frac{6}{14} * 0.9 + \frac{4}{14} * 0.815 = 0.904$$

Gain(Age) $= 0.925 - 0.904 = 0.021$

Student

$$E_{no} = -\left(\log\left(\frac{4}{7}\right) * \left(\frac{4}{7}\right) + \log\left(\frac{3}{7}\right) * \left(\frac{3}{7}\right)\right) = 0.983$$

$$E_{yes} = -\left(\log\left(\frac{6}{7}\right) * \left(\frac{6}{7}\right) + \log\left(\frac{1}{7}\right) * \left(\frac{1}{7}\right)\right) = 0.591$$

$$E_{student} = \frac{7}{14}(0.983 + 0.591) = 0.787$$

Gain(student) $= 0.9 - 0.787 = 0.113$

Credit

$$E_{fair} = -\left(\log\left(\frac{6}{8}\right) * \left(\frac{6}{8}\right) + \log\left(\frac{2}{8}\right) * \left(\frac{2}{8}\right)\right) = 0.815$$

$$E_{low} = -\left(\log\left(\frac{3}{6}\right) * \left(\frac{3}{6}\right) + \log\left(\frac{3}{6}\right) * \left(\frac{3}{6}\right)\right) = 1$$

$$E_{student} = \frac{8}{14} * 0.815 + \frac{6}{14} * 1 = 0.894$$

Gain(student) $= = 0.9 - 0.894 = 0.03$

بنابراین ریشه‌ی درخت age می‌شود.

فرایند بالا را برای پیدا کردن ادامه‌ی درخت ادامه میدهیم.

$$E(S_{youth}) = -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log(2/5) = 0.97$$

Student

$E_{yes} = 0$
$E_{no} = 0$
$E_{student} = 0$
$\text{Gain(student)} = 0.97 - 0 = 0.97$

بیشتر از این مقدار بهره نمیتوانیم داشته باشیم، بنابراین در همین مرحله ریشه‌ی درخت بعدی مشخص میشود و دیگر شاخه‌ی youth را ادامه نمی‌دهیم.
در ادامه شاخه‌ی senior را ادامه میدهیم.

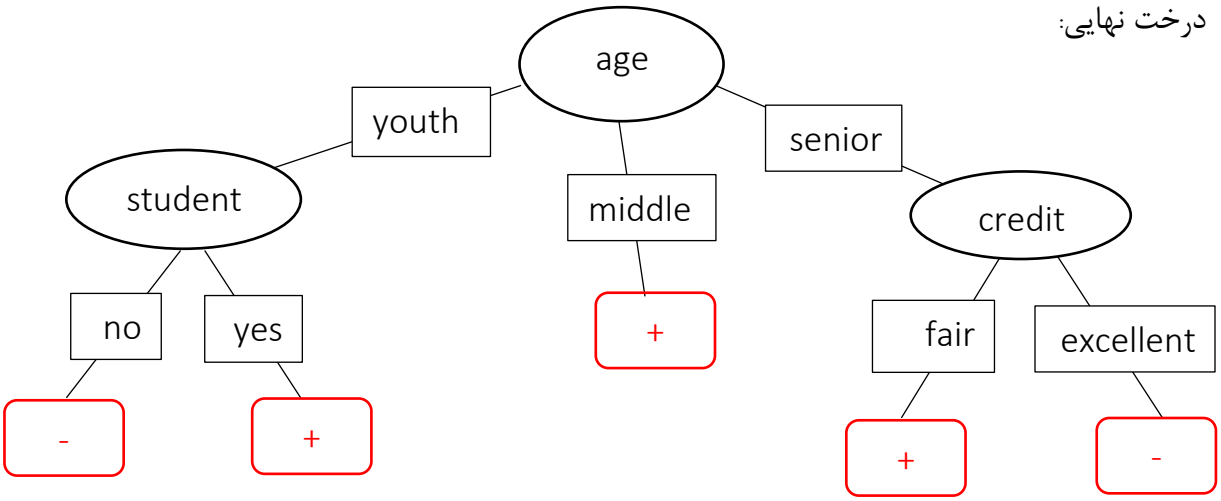
$$E(S_{senior}) = -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log(2/5) = 0.97$$

Credit

$E_{fair} = 0$
$E_{exellent} = 0$
$E_{credit} = 0$
$\text{Gain(credit)} == 0.97 - 0 = 0.97$

credit برای ریشه زیر درخت انتخاب می‌شود و نیاز به ادامه دادن نیست. در ادامه درخت ساخته شده مشاهده میشود.

درخت نهایی:



با استفاده از درخت ساخته شده در بالا برچسب هر نمونه به صورت زیر است:

X_1 = (age = youth, income = high, student = yes, credit = fair) => +

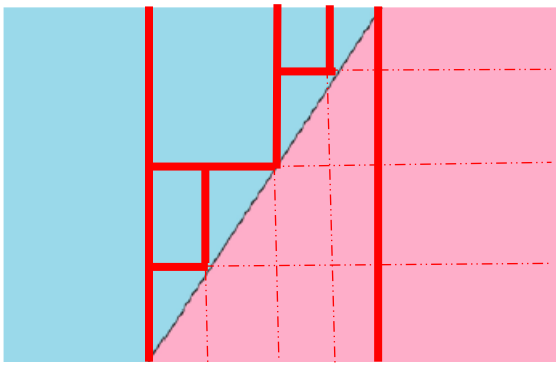
X_2 = (age = senior, income = low, student = no, credit = excellent) => -

X_3 = (age = middle-aged, income = medium, student = no, credit = fair) => +

موارد زرد شده در تعیین برچسب موثر بوده اند.

سوال هشتم

خیر نمیتوان با عمق محدود این مرز را ترسیم کرد. زیرا در درخت تصمیم هر گره تصمیم گیری فضای داده‌ها را با یک خط موازی محورها تقسیم می‌کند. برای چنین خطی هرچقدر هم عمق درخت را زیاد کنیم باز میزانی خطا داریم. همانطور که در شکل X مشاهده میکنیم این فرایند تا تعداد زیادی عمق ادامه پیدا می‌کند.



سوال نهم

در این الگوریتم داده‌ها را به مجموعه‌های تصادفی کوچکتر تجزیه میکنیم (تعداد این مجموعه‌ها کم نباید باشد)، سپس روی هر کدام از این مجموعه‌ها یادگیری را انجام می‌دهیم و درنهایت نتایج را ادغام می‌کنیم. استفاده از این الگوریتم در درخت تصمیم باعث می‌شود که بتوانیم برای هر مجموعه یک درخت مجزا با عمق محدود داشته باشیم و نتیجه‌ی نهایی از آن نتیجه‌ای است که بیشترین بار تکرار شده است در کل درخت‌ها.

بخش وکا

سوال اول

با توجه به تصویر زیر مقادیر ماتریس درهم ریختگی برابر با : TP = 9 , TN = 46 , FP = 8 , FN = 23 است.

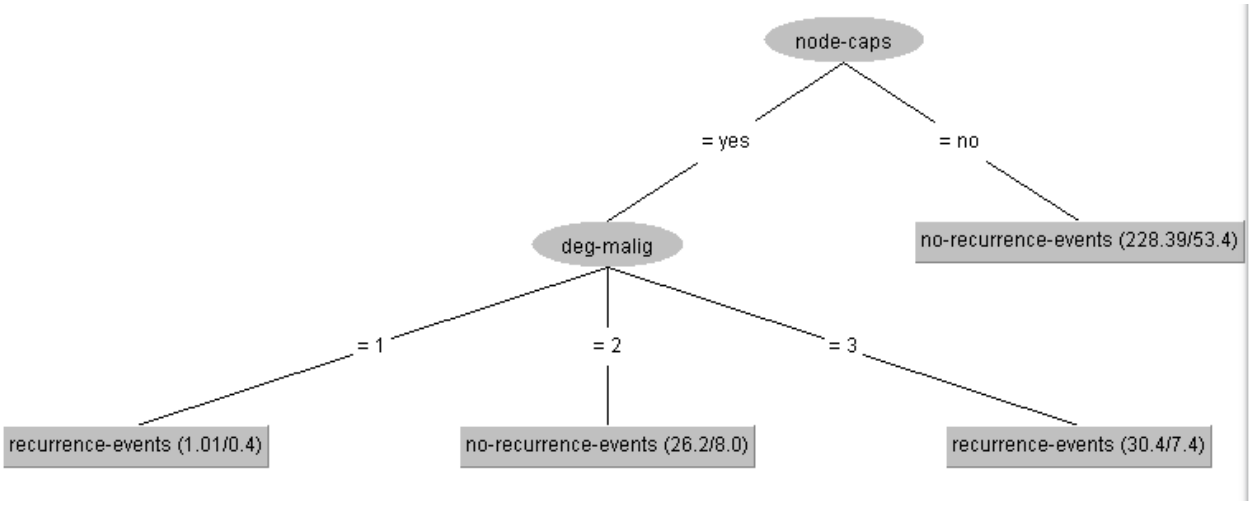
```
=== Confusion Matrix ===

  a  b   <-- classified as
 46  8 |  a = no-recurrence-events
 23  9 |  b = recurrence-events
```

بنابراین دقت و پوشش به صورت زیر هستند:

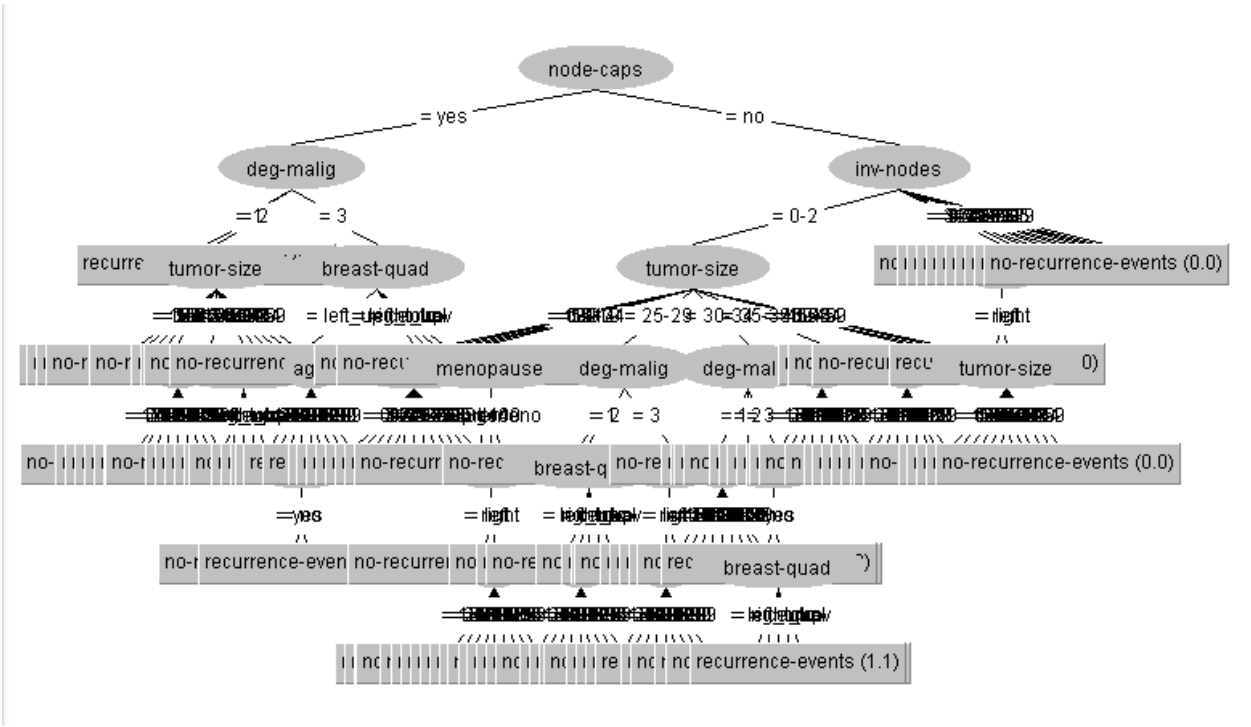
Precision = TP / TP + FP = 0.52 recall = TP / TP + FN = 0.28

F1measure = 0.36

تصویر درخت به صورت زیر است:



ب) unpruned یعنی درخت ساخته‌شده، هرس نشود. در واقع در حالتی که عمق درخت زیاد شود و به اصطلاح overfitting رخ دهد، با هرس کردن درخت بعضی از شاخه‌ها حذف شده و در ریشه‌ی مربوطه احتمال آنها نوشته میشود. انتظار می‌رود که با فعال کردن این گزینه عمق درخت بیشتر شود. در ادامه شکل مربوط به درخت جدید را میبینیم.



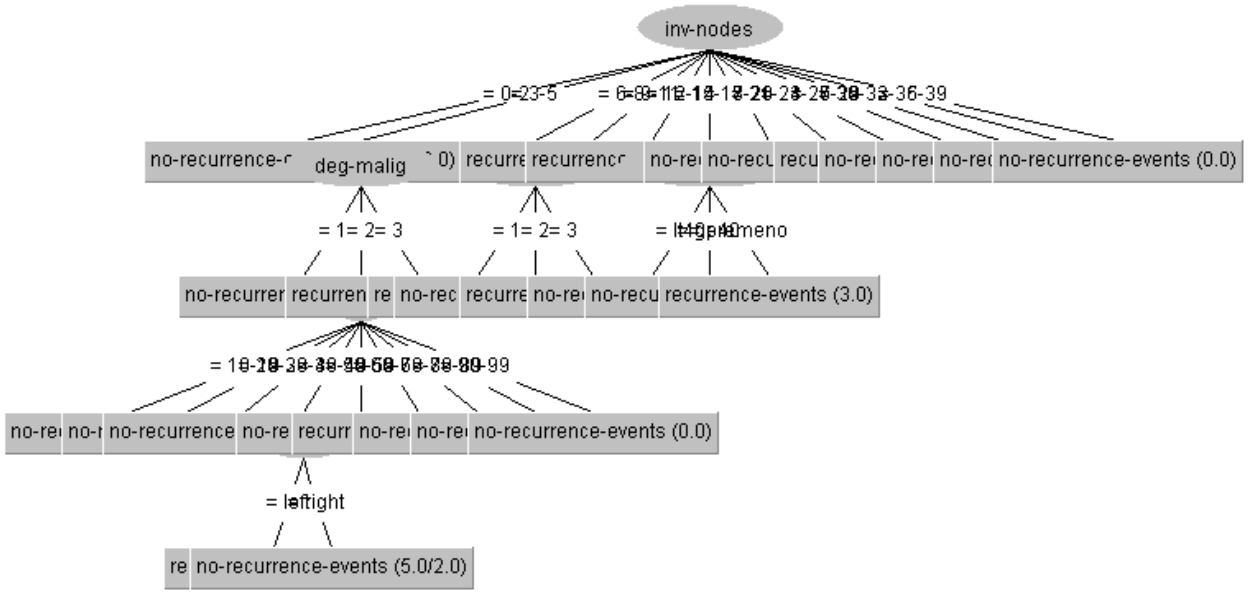
```
=== Confusion Matrix ===

 a  b   <-- classified as
41 13 |  a = no-recurrence-events
20 12 |  b = recurrence-events
```

TP = 12 , FP = 13 , TN = 41 , FN = 20

Recall = 0.37 precision = 0.48 F1measure = 0.42

در این حالت تعداد تشخیص درست کلاس مثبت یعنی کلاس b بیشتر شده است و این به دلیل اورفیت شدن مدل است. چون درخت هرس نشده. همچنین معیار f1 نیاز بهتر شده است.

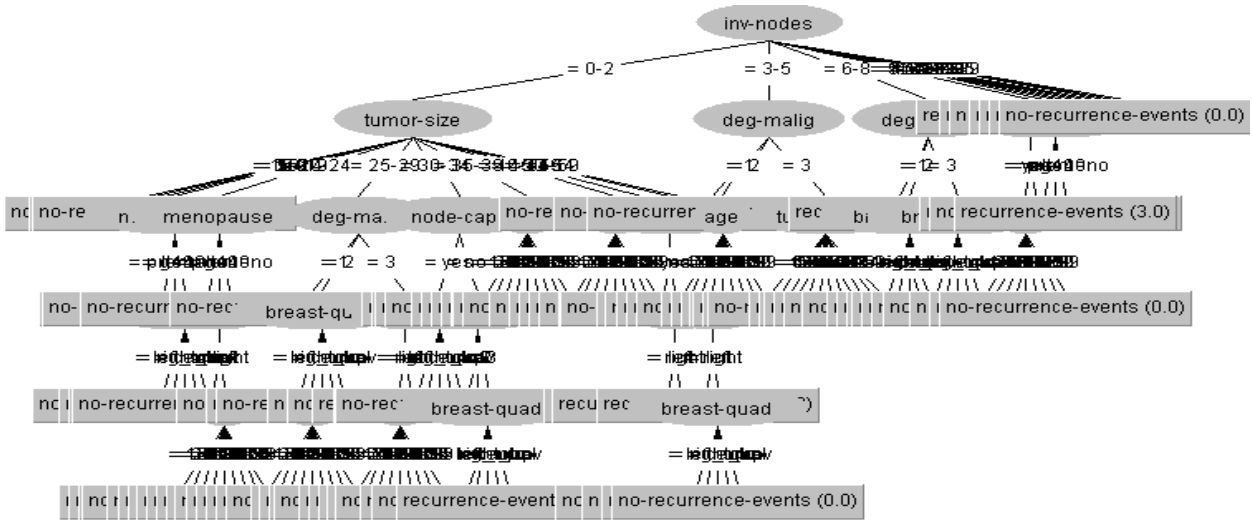
ج) برای مورد اول شکل درخت :




```
=== Confusion Matrix ===

 a  b   <-- classified as
47  7 |  a = no-recurrence-events
23  9 |  b = recurrence-events
```

TP = 9 , TN = 47 , FP = 7 , FN = 23 , precision = 0.56 , recall = 0.28 , F1measure = 0.37

برای مورد دوم شکل درخت :




```
=== Confusion Matrix ===

  a  b   <-- classified as
 39 15 |  a = no-recurrence-events
 19 13 |  b = recurrence-events
```

TP = 13 , FP = 15 , TN = 39 , FN = 19

Recall = 0.40 precision = 0.46 F1measure = 0.43

همانطور که مشاهده می‌کنیم در هر دوحالت ریشه‌ی درخت عوض شده است.

در حالت اول که از بیش برازش جلوگیری کرده است با وارد کردن نویز مدل بهتر یادگیری را انجام داده. درواقع مدل داده ها را حفظ نکرده است.

اما در حالت دوم که بیش برازش داشته ایم و مدل داده‌ها را حفظ کرده است با وارد کردن نویز معیارهای ارزیابی کاهش یافته اند. دلیل این امر این است که دراین حالت نویز نیز حفظ شده است و چون در داده‌های تست نویز وجود نداشته است عملکرد مدل خوب نشده است.

بخش پیاده‌سازی

سوال اول

در ابتدا برای درست خوانده شدن فایل داده، یک سطر به اول فایل اضافه شد. سطر اضافه شده به صورت زیر است.

```
0,1,2,3,4,5
```

اگر این کار انجام نمیشد سطر اول داده‌ها به عنوان نام ستون در نظر گرفته می‌شد.

سپس داده‌ها با عملیات split از یک ستون به ۶ ستون گسترده شدند.

برای پیش‌پردازش داده‌ها، در ابتدا مقادیر نامشخص با میانگین مقادیر موجود پر شد. یعنی در هر ستون میانگین داده‌های موجود حساب شد و به جای علامت سوال‌ها قرار گرفت. همچنین در بعضی از نمونه ها علامت سوال وجود نداشت و داده‌ای نیز نبود. در آن خانه‌ها نیز میانگین کل قرار گرفت.

الف) برای این قسمت سوال برای هر قسمت از ۱۰ تا بخش cross_validation تمام معیارها محاسبه شده است و سپس از آنها میانگین گرفته شده است.

در یک بار اجرای برنامه تمام پارامترها، یعنی kهای مختلف اجرا شده اند و معیارها و زمان کل ثبت شده است.

(تاکید میکنم این زمان برای تمام kها همزمان است نه یک بار اجرا به ازای هر k. همچنین این زمان، شامل زمان محاسبه‌ی فاصله‌ها نیز می‌باشد.)

در ادامه نتایج را میبینیم.

```
--- 264.9056763648987 seconds ---
accuracy k = 1
0.7503329037800687
TN = %i | TP = %i | FN = %i | FP = %i 40.1 32.0 12.4 11.6
accuracy k = 3
0.7857603092783505
TN = %i | TP = %i | FN = %i | FP = %i 41.8 33.7 10.7 9.9
accuracy k = 5
0.7992697594501718
TN = %i | TP = %i | FN = %i | FP = %i 41.9 34.9 9.5 9.8
accuracy k = 7
0.8034149484536082
TN = %i | TP = %i | FN = %i | FP = %i 42.1 35.1 9.3 9.6
accuracy k = 15
0.8075386597938146
TN = %i | TP = %i | FN = %i | FP = %i 40.9 36.7 7.7 10.8
accuracy k = 30
0.7939862542955327
TN = %i | TP = %i | FN = %i | FP = %i 39.8 36.5 7.9 11.9
```

همانطور که مشاهده می‌کنیم بیشترین دقت به $k = 15$ تعلق دارد. بنابراین قسمت بعد سوال را با همین k ادامه میدهیم.

ب)

برای فاصله‌ی منهتن :

```
accuracy k = 15
0.8033934707903778
TN = %i | TP = %i | FN = %i | FP = %i 40.8 36.4 8.0 10.9
```

برای فاصله‌ی کسینوسی:

```
accuracy k = 15
0.7793921821305841
TN = %i | TP = %i | FN = %i | FP = %i 39.4 35.5 8.9 12.3
```

برای فاصله‌ی اقلیدسی نیز در قسمت قبل محاسبه شد.

ج)

همانطور که در جدول زیر مشاهده می‌کنیم میزان دقت بین الگوریتم نوشته شده و توابع آماده تفاوت چندانی وجود ندارد. تنها تفاوت شایان توجه در زمان اجرا است.

K=1 --- 0.010099649429321289 seconds --- fp fn tp tn 11 11 39 35 accuracy 0.7731958762886598	K=3 --- 0.010469436645507812 seconds --- fp fn tp tn 9 14 36 37 accuracy 0.7628865979381443
K=5 --- 0.009360790252685547 seconds --- fp fn tp tn 10 13 37 36 accuracy 0.7628865979381443	K=7 --- 0.009831905364990234 seconds --- fp fn tp tn 9 13 37 37 accuracy 0.7731958762886598
K=15 --- 0.008793115615844727 seconds --- fp fn tp tn 9 10 40 37 accuracy 0.8041237113402062	K=30 --- 0.009730339050292969 seconds --- fp fn tp tn 11 10 40 35 accuracy 0.7835051546391752

سوال دوم

بر روی داده‌های آموزش :

K	7	15	30	5	3	1

MSE	۶۳.۶۳	۷۰.۲۴	۷۵.۹۵	۵۱.۴۰	۳۵.۱۶	۳.۱۰

همانطور که از جدول بالا پیداست $k = 1$ کمترین ارور را دارد. برای داده‌های تست با یک همسایه نزدیک، ارور حاصل ۶۱.۲۵ است.

اما برای حالت تست در ۵ نزدیک‌ترین همسایه، ارور برابر با ۵۶.۸۳۶ می‌شود. یعنی در حالت تست بهترین k برابر با ۵ است و یک نیست. دلیل این امر نیز مشهود است. در فاز آموزش چون از خود داده‌ها استفاده میشود بهترین k برابر با یک می‌شود که همان داده‌ی آموزش است و فاصله نزدیک به صفر بدست می‌اید. اما در فاز تست بهتر است k عددی بزرگتر از یک باشد و میانگین فاصله‌ها در نظر گرفته شود.

نکته : در حالت آموزش چون در محاسبه‌ی یک نزدیک‌ترین داده، به داده‌ی مورد بحث همان داده است پس توقع داریم که کمترین خطا را داشته باشیم یعنی حتی بسیار نزدیک به صفر.

سوال سوم

در این سوال، مقادیر ناموجود با میانگین مقادیر موجود پر شده است.

به ابتدای هر دو فایل تست و آموزش یک سطر با مقادیر ۰ تا ۱۰ اضافه شده تا سطر اول داده‌ها به عنوان نام ویژگی در نظر گرفته نشود.

در داده‌های آموزش دو نمونه وجود داشت که مقدار None داشتند، آن مقادیر نیز با میانگین پرشده اند.

برای استفاده از تابع آماده می‌بایست اعداد داخل آرایه باشند، اما در هنگام خواندن از فایل این مقادیر به صورت رشته ذخیره شده یودند. به همین جهت با تابع زیر فرمت تمامی مقادیر ارایه به عدد تبدیل شده است.

```
train_data = train_data.apply(pd.to_numeric, errors='coerce')
test_data = test_data.apply( pd.to_numeric, errors='coerce' )
```

در ستون یازدهم مقادیر به صورت زیر هستند:

Class: 2 for benign, 4 for malignant

بنابراین مقدار ۲ کلاس منفی و مقدار ۴ کلاس مثبت در نظر گرفته شده و ماتریس درهم ریختگی به صورت زیر محاسبه شده است:

```
for i in range(predicted_value.size):
    if (predicted_value[i] == 2 and test_data.iloc[i, 10] == 2):
        TN = TN + 1
    if (predicted_value[i] == 2 and test_data.iloc[i, 10] == 4):
        FN = FN + 1
    if (predicted_value[i] == 4 and test_data.iloc[i, 10] == 2):
        FP = FP + 1
    if (predicted_value[i] == 4 and test_data.iloc[i, 10] == 4):
        TP = TP + 1
```

دقت نیز به صورت زیر محاسبه شده:

```
print((TN + TP) / predicted_value.size)
```

خروجی حاصل به شکل زیر است:

```
accuracy is :
0.92
TP TN FN FP
46 138 12 4
```