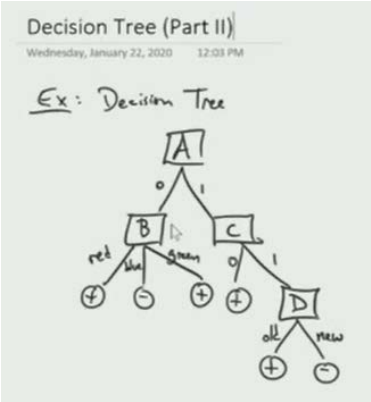
بسم الله الرحمن الرحيم

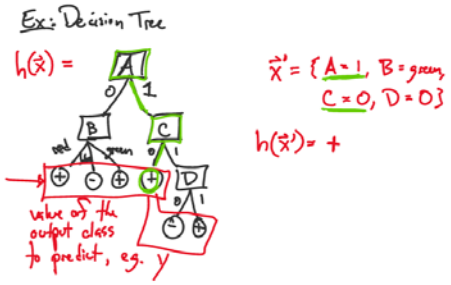
المحاضره ديه هنتكلم علي أول الجورذم هيبقا مفيد في ال decision trees .. المهم ان HW2 نزل .. فابدا فيه علي طول .. تعال نفكر في ال decision tree كإنها hypothesis .. تعال نرسم مثال .. ال Decision tree اللي هنشتغل عليها بيتكون من شوية Attributes : بص ع الرسمه



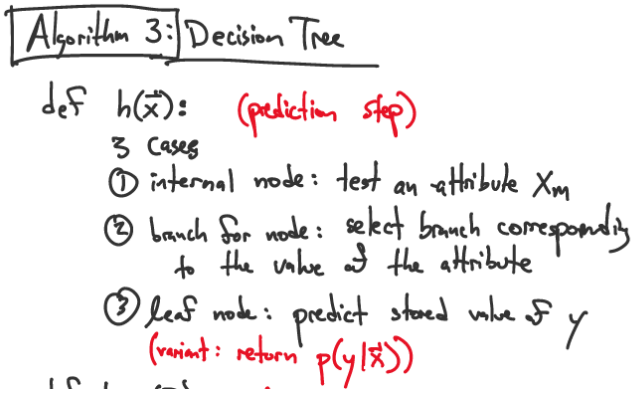
المهم احنا هنشتغل علي الشجره اللي فوق ديه .. تعال نشوف ايه اللي هيحصل لو دخل مثال جديد علي الشجره ديه .. هنسميها إكس برايم .. هيبقا ليه شوية قيم لل Attributes اللي عندنا ...

$$\vec{x}' = \{A=1,\ B=\text{green},\ C=0,\ D=0\}$$

المهم بيقولك احنا في الشجره بتاعتنا ... عند ال A .. احنا عندنا h(x) و ديه فانكشن بت map ال x values ل y values .... فاحنا هنبدأ في المثال الجديد اللي عندنا من أول A .. هنلاقي قيمتها 1 .. وبعدها نخش علي C .. هل القيمه صفر ولا واحد .. هتلاقي انها صفر .. فهترجع "+" ... فهتلاقي ال return



ديه كدا الفانكشن اللي هي ال hypothesis .. احنا مهتمين بقا هي ازاي تتعلم .. تعال نشوف ال algorithm 3 اللي عندنا وهو ال decision tree algorithm ... تعال نبدأ نعرّف ال h(x hat) .. هنا عندك 3 حالات -> اول حاله هي لو internal node .. وديه عباره عن أي حاجه موجوده ماعدا ال leaf node .. هنا بقي انت هت test شوية attributes Xm في ال node ديه .. تاني حاله هي ان لو في branch لل node اللي عندنا ... هنختار ال branch المقابل ل القيمه لل mth attribute اللي هي ال Xm يعني .. وبعدين نقوم ماشيين في الفرع ده .. في الحاله التالته .. وهي ان بيبقا عندنا leaf node .. وهي إننا ن predict stored value of y ... خد بالك ان الحاله التالته مختلفه شويه .. وهي إنك مش بترجع بس ال + او - .. انت هترجع probability distribution ..



تعال نفهم هو ليه مستخدمين ال variant الأخير في الحاله رقم 3 .. بص ع المثال ده .. احنا بنبص علي حالات من النسوان الحامل .. بنبص علي ال medical record .. و لكل مريض عندنا different attributes .. فاحنا عندنا ال attributes ديه واحنا بنحاول ن predict ال C-Section Risk ..

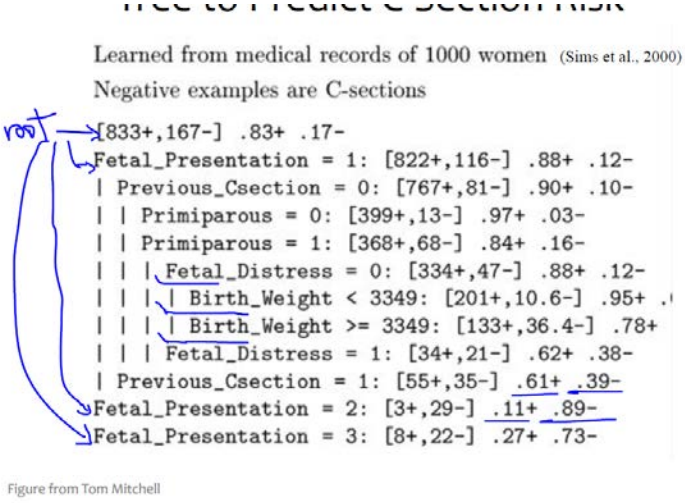


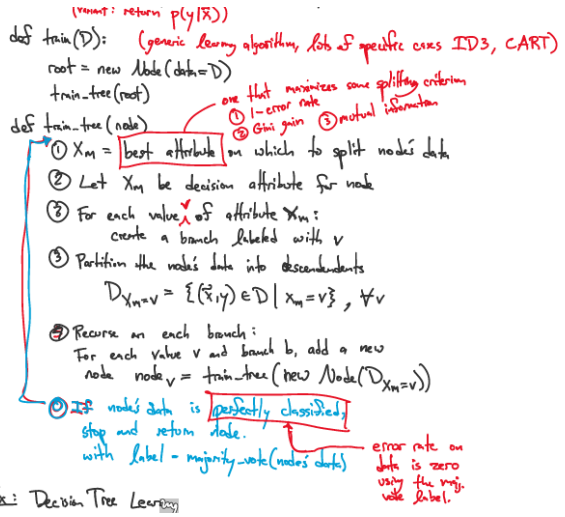
Figure from Tom Mitchell

9

المهم تعال نعمل ترينج ألجورذم .. وده هنا هنعرف recursive algorithm .. انت عندك داتا سيت D .. احنا هنا بس بنعرّف generic version of the algorithm ... بس في حالات خاصه من ال algorithm ده زي مثلاً .... ID3, CAR, ... بس في الآخر الفكره واحده و الاختلاف مش بيبقا كبير يعني ..

أول خطوه هي إننا هنبدأ ال root node و ديه هتاخد data = D .. هتجيلها ال Full trainng set D . وبعدين تبدأ ال recursion implementation .. return(train_tree(root)) ... الفانكشن اللي اسمها train_tree ديه اللي هتعمل ال recursion علي ال root ديه .. تعال نشوف الفانكشن ديه بتشتغل ازاي ...

أول خطوه إنك تحسب attribute Xm .. و ده اللي هو أحسن attribute نقدر ن split عليه ال nodes data .. فال node ديه أوبجكت ... جواها شوية data .. تاني خطوه هو إنك أول ما انت خلاص بقي عندك ال Xm .. خليه هو ال decision attribute لل node الحاليه .. وبعدين انت عاوز تبني شجره .... فهتقول .. لكل قيمه من قيم ال attribute Xm .. هنحتاج نعمل branch ليه اسم بإسم القيمه اللي داخلين من ناحيتها .. رابع خطوه هو انك تقسّم ال node's data لل branches اللي داخلين عليها اللي هو ال children بتوع ال current node .. فهتلاقي نفسك بتعرّف داتا سيت اسمها D .. بس المره ديه بتتكون من X and Y pairs بس دول اللي موجودين جواال node الحاليه .. بحيث إن قيمة ال Xm هي قيمة ال value = ... كدا نخش علي خطوة ال recursion .. recurse on each branch ... لكل قيمة من قيم ال values اللي موجوده في ال branch الحالي .. هنعمل new node .. وديه هتبقا مخصصه ل value الحاليه يعني .. وديه هتاخد القيمه بتاعت ال recursion بتاع ال train_tree تاني .. بس المره ديه هنبعتلها ال new_node .. والداتا بتاعت ال node اللي هنبعتها تاني .. هي ال sub dataset اللي كوّناها فوق .... فاضل مشكله واحده .. احنا هنقف امتي؟؟ .. فاحنا محتاجين base case .. اللي هي هتقولك ده ال situation اللي عندو هتخرج من ال recursion .. في الحاله ديه .. هتقول لو النود داتا خلاص بقت perfectly classified .. اخرج .. بس وانت بتخرج ابعتلي ال label بتاع ال majority vote علي ال current node's data .. بس هنا في مشكله إيه معني ان ال node data be perfect classified? .. perfect classification معناها لما يكون ال error rate بصفر . اللي هو يا كلو + يا كلو – ده اللي هو ال majority vote perfectly classified



المهم في الصوره اللي فوق في حاجه مهمه .. وهي إنك ازاي تختار ال best attributes .. اختار اللي ت maximize 1- error rate .. او ت maximize Gini gain .. او انك ت maximize mutual information ..

تعال ناخد مثال: ..

Decision Tree Learning Example

Dataset:

Output Y, Attributes A, B, C

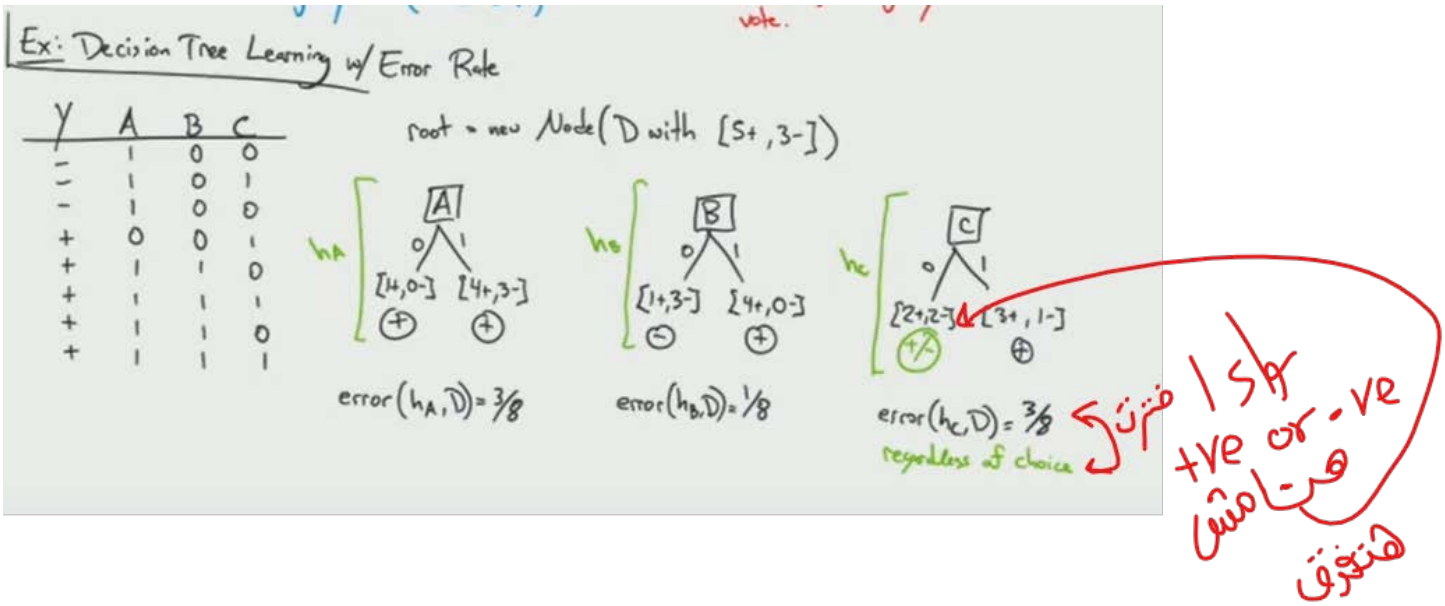
Y	A	B	C
-	1	0	0
-	1	0	1
-	1	0	0
+	0	0	1
+	1	1	0
+	1	1	1
+	1	1	0
+	1	1	1

In-Class Exercise

Using **error rate** as the splitting criterion, what decision tree would be learned?

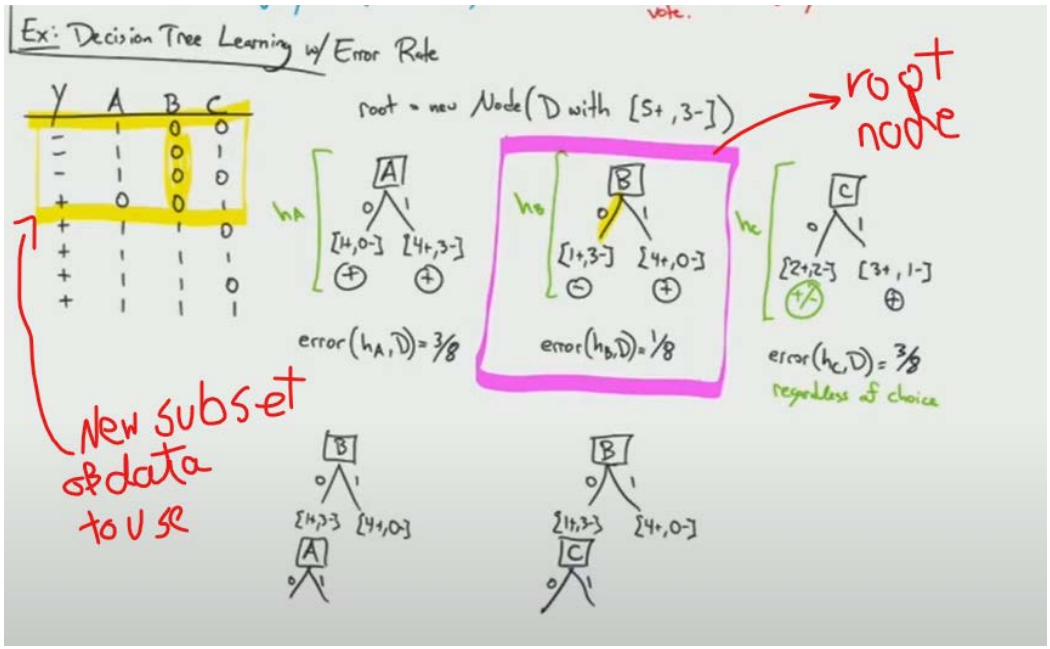
11

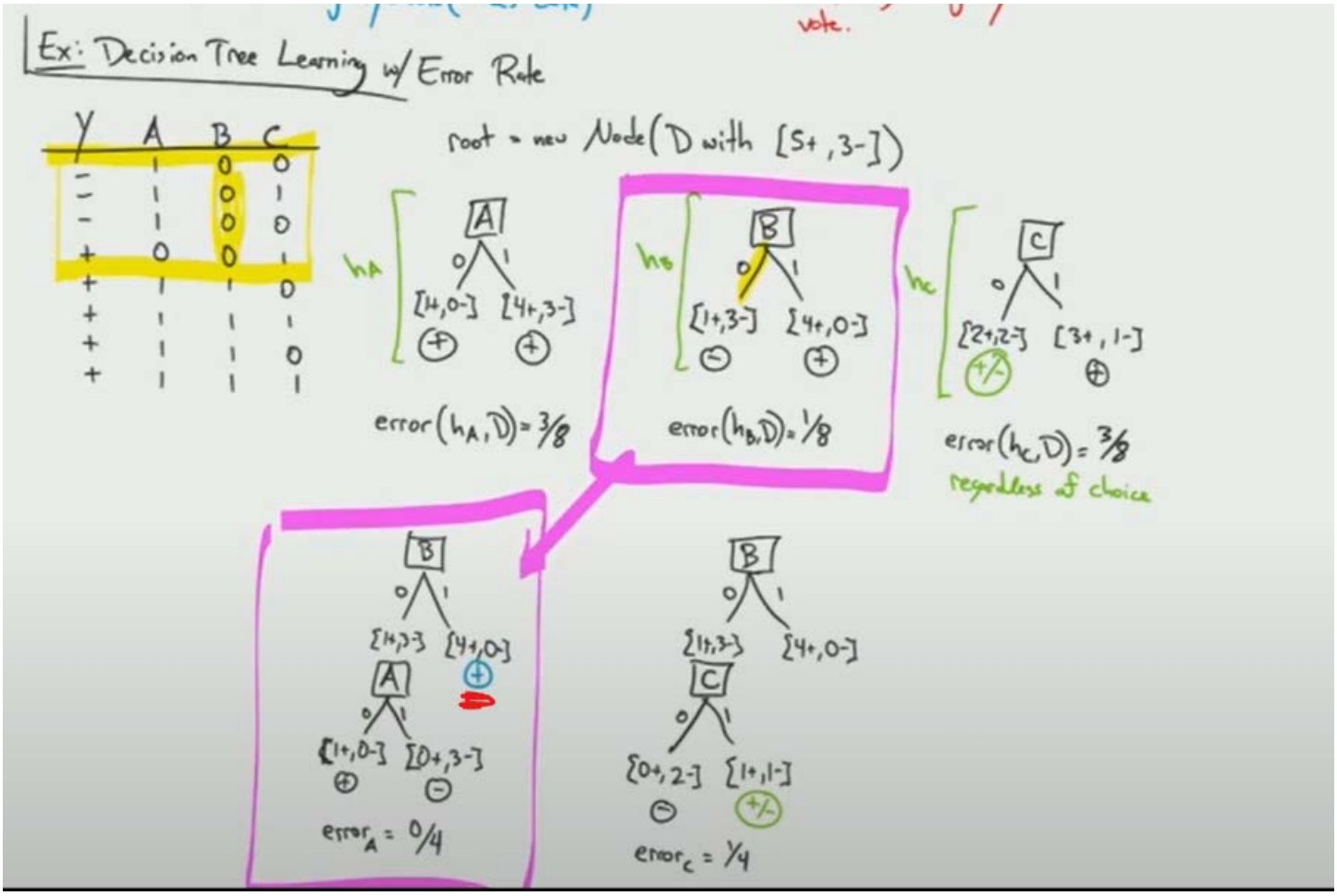
احنا عاوزين نحدد ال root node .. وده هيبقا new node object .. وهنا هيبقا عندنا ال DataSet كلها .. السؤال هنا .. فيها كام واحده + وكام واحده - .... هتلاقي عندك 5 بوزيتيفو 3 نيجاتيف .. انت عندك attributes 3 ABC .. و دول كلهم binary .. لكل split عندنا .. هنقول مثلاً . لو A كانت بصفر .. انت عندك واحده + و معنديش - ... لو ال A كانت بواحد .. يبقا انت عندك 3 ve– و 5 ve+ .. نفس الكلام بالنسبة ل B ..



المهم انك هتختار أقل error .. احنا كدا هن split علي B ... ويلا بينا علي ال recursion .. احنا لما بن recurse .. بن recurse علي multiple options .. فا يإما هنفصل علي علي ال A أو هنفصل علي ال C .. المهم انك تاخد بالك من حاجه مهمه ... انت لما بتبدأ ال recursion انت مش بتشتغل علي نفس الداتا سيت .. انت بتشتغل علي subset من الداتا سيت اللي كانت في الأول ..

مثلاً .. لما قلنا إن لو ال B كانت بصفر .. فانت دخلت في البرانش بتاع الصفر ... فدلوقت الداتا سيت بتاعتك اللي هتشتغل عليها هي .. اللي متلوونه باللون الاصفر .





Decision Tree Learning

- *Definition*: a **splitting criterion** is a function that measures the effectiveness of splitting on a particular attribute
- Our decision tree learner **selects the "best" attribute** as the one that maximizes the splitting criterion
- Lots of options for a splitting criterion:
  - error rate (or *accuracy* if we want to pick the tree that *maximizes* the criterion)
  - Gini gain
  - Mutual information
  - random
  - …

13

تعال نشوف مثال:

Decision Tree Learning Example

Dataset:
Output Y, Attributes A and B

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1

In-Class Exercise

Which attribute would **error rate** select for the next split?

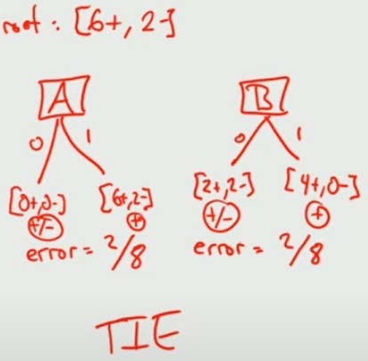
1. A
2. B
3. A or B (tie)
4. Neither

14

Decision Tree Learning Example

Dataset:
Output Y, Attributes A and B

Y	A	B
-	1	0
-	1	0
+	1	0
+	1	0
+	1	1
+	1	1
+	1	1
+	1	1



Givan a random variable Y over K classes {1,2,..,k} ... Gini Impurity هنعرّف حاجه اسمها .. ازاي نعالج التعادل اللي عندنا ده؟

هنقول ان التعريف بتاع Gini Impuity .. هو عباره عن G(y) .. مجموع حاصل ضرب احتمالية إن الY=k و ال y!=k ... من اول ال k=1 -> K

تعال نفكّر في الآتي .. consider the case where y is the outcome of a weighted dice roll

P(Y=k) is the prob of landing on side k

P(Y!=k) is the prob of landing on any other side besides k

اللي عاوزين نعملو اننا ن conceptualize a gain

في اللعبه هنا هدفنا اننا ن predict the next dice roll .. و مديك ال weights of the die ... فانت عارف من الاول احتمالية كل side بكام .. ايه الحل بتاعك في الحاله ديه ... ازاي هتلعب اللعبه ديه ..

أول حل ... توقّع ال most probable side every time .. في الحاله ديه ... ال expected error rate .. تقدر تقول إنها 1-P(y=most probable side) اللي هي P(Y!=Y*) .. ده عباره عن Summation over all the values k != y*

.. تاني حل .. هي إنك ت roll another die w/some weight on sides and predict whatever it lands on expected error rate... الحقيقه مفهمتش الحته ديه .. ابقا اسمع آخر 10 دقايق تاني ..