Classification in Data Mining
التصنيف: خوارزمية C4.5

1. مقدمة:

التصنيف Classification: هو بناء نموذج Model يسمى classifier يتم تدريبه على أمثلة مختلفة(instance/example/record) في مجموعة بيانات(database/dataset) ووضعها في تصنيفات محددة(classes/categories) ومعروفة مسبقاً من خلال تحليل خصائص(سمات/صفات) الأمثلة (Attribute/feature) واستخدامها لبناء قرار عن الصنف الذي ينتمي إليه مثال ما. كتصنيف الصور والأصوات وتشخيص الأمراض وغيرها، والتصنيف من تقنيات تعلم الآلة (Machine learning) بإشراف(supervised learning) المستخدمة في التنقيب عن المعطيات.

2. أنواع خوارزميات التصنيف:

توجد العديد من خوارزميات التصنيف والتي تستخدم في تصنيف البيانات بحسب خصائصها، ومن بين هذه الخوارزميات:

1- خوارزميات أشجار القرار Decision Tree:
2- الاحتمالات البسيطة Naive Bayes:
3- الغابات العشوائية Random Forest:
4- آلات الدعم المتجهي Support Vector Machines (SVMs):
5- الجيران الأقرب K-Nearest Neighbor (KNN):
6- النحدار الخطي Logistic Regression:
7- الشبكات العصبونية Neural Networks:

3. بنية البيانات:

- تتألف مجموعة البيانات من عدة أمثلة تنتمي إلى أصناف متنوعة.
- كل مثال E يتألف من قيمة الصنف التابع له، وشعاع من الخصائص عددها m لها الشكل:

$$(A^1, \dots, A^i, \dots, A^m)$$

- كل خاصية لها قيمة إما متقطعة أو مستمرة. وبالتالي فإن المثال يعرّف كما يلي:

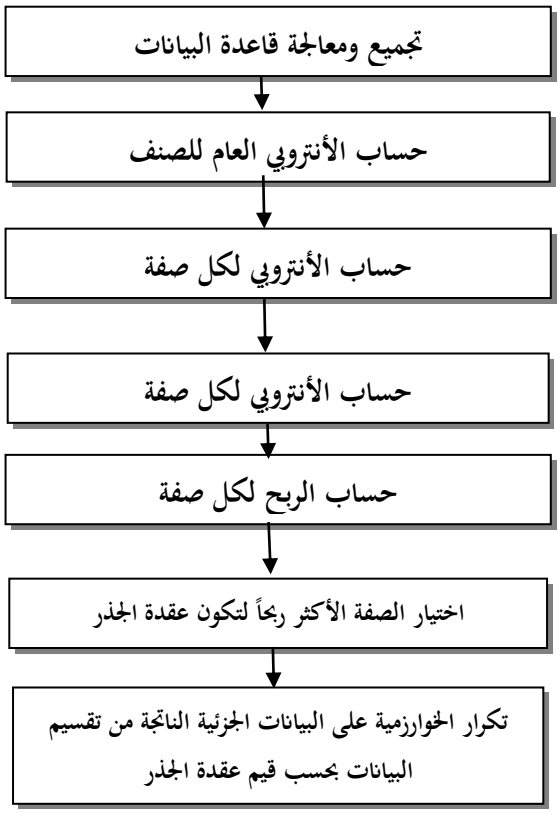
$$E = (A^1 = V_E^1, \dots, A^i = V_E^m, \dots, A^m = V_E^m, \text{Class} = C_E).$$

مثال: مجموعة أمثلة تدريب مستخدمة إمكانية لعب الغولف حسب الطقس:

Outlook	Temperature	Humidity	Windy	Class (Play)
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	78	false	yes
rain	70	96	false	yes
rain	68	80	false	yes

4. خوارزمية C4.5:

تعتبر من الخوارزميات الهامة المستخدمة في عملية التصنيف، وفيها يتم بناء نموذج على شكل شجرة قرار تعبر عن البيانات المدروسة، تستخدم خوارزمية C4.5 ما يسمى بمفهوم ربح المعلومة وذلك من خلال تطبيق مفهوم الأنتروبي. يبين الشكل التالي المخطط النهجي لخوارزمية C4.5:



1) حساب الأنتروبي العام لمجموعة بيانات التدريب T :

$$H(T) = -\sum_{j} Pj * \log_2(Pj) \qquad (1)$$

k: تمثل عدد الأصناف (j=1 --> k).

Pj: احتمال تكرار الصنف،

ويساوي (عدد أمثلة الصنف/عدد الأمثلة الكلي) وقيمته بين 0 و1.

2) حساب الأنتروبي للصفات:

$$Hs(T) = \sum_{i=1}^{k} Pi * Hs(Ti) \qquad (2)$$

k: تمثل عدد قيم الصفات.

Pi: احتمال تكرار قيمة الخاصية المحددة،

ويساوي (عدد مرات قيمة الخاصية/عدد الأمثلة الكلي).

Hs(Ti): تطبيق العلاقة (1) على أصناف قيم الصفة الواحدة.

S الخاصية المختارة.

3) حساب الربح لكل صفة وفق العلاقة التالية:

$$Gain(S) = H(T) - Hs(T) \qquad (3)$$

وتكرر العملية من أجل كل صفة حتى يتم تحديد صاحب الربح الأكبر.

–

4) أنواع العقدة في شجرة القرار المولدة:

1. عقدة ورقة leaf node: تعبِّر عن قاعدة نهائية.
2. عقدة قرار(داخلية) decision node: تحدد اختباراً ما على قيمة خاصية مفردة مع فرع وحيد وشجرة جزئية لكل ناتج محتمل للاختبار.

ومن ثم يمكن استعمال الشجرة الناتجة لتصنيف مثال جديد بدءاً بعقدة الجذر root node مروراً بالمسارات وصولاً إلى عقدة ورقة تمثل صنفاً ينتمي إليه هذا المثال.

5) مثال:

لتكن لدينا مجموعة تدريب T بسيطة مكونة من 14 مثال، موصوفة من خلال ثلاث خصائص دخل وتنتمي إلى أحد الصنفين Calss1 أو Class2. يبين الجدول التالي هذه المجموعة.

Attribute1	Attribute2	Attribute3	Class
A	70	True	Class1
A	90	True	Class2
A	85	False	Class2
A	95	False	Class2
A	70	False	Class1
B	90	True	Class1
B	78	False	Class1
B	65	True	Class1
B	75	False	Class1
C	80	True	Class2
C	70	True	Class2
C	80	False	Class1
C	80	False	Class1
C	96	False	Class1

من الملاحظ أن تسعة أمثلة تنتمي إلى الصنف الأول وبقية الأمثلة تنتمي إلى الصنف الثاني، وعليه فإن الأنتروبي العام قبل التقسيم يكون:

$$H(T) = -\ 9/14 \log2(9/14) - 5/14 \log2(5/14) \quad = 0.940 \text{ bits.}$$

باستعمال الخاصية (الصفة) الأولى لتفريق المجموعة T إلى ثلاث مجموعات جزئية، حيث الاختبار x1 يمثل الاختيار بين ثلاث قيم A, B, or C. نجد أن المعلومات الناتجة هي:

$$H_{x1}(T) = 5/14 \ (-\ 2/5 \log2(2/5) - 3/5 \log2(3/5))$$
$$+ 4/14 \ (-\ 4/4 \log2(4/4) - 0/4 \log2(0/4)$$
$$+ 5/14 \ (-\ 3/5 \log2(3/5) - 2/5 \log2(2/5) = 0.694 \text{ bits}$$

ويكون ربح المعلومات من أجل الاختبار x1 (الخاصية الأولى attribute1):

$$Gain(x1) = 0.940 - 0.694 = 0.246 \text{ bits.}$$

إذا كان الاختبار مبنياً على الخاصية الثالثة، حيث x2 يمثل اختيار واحدة من قيمتين True or False.

$$H_{x2}(T) = 6/14 \ (-\ 3/6 \log2(3/6) - 3/6 \log2(3/6))$$
$$+ 8/14 \ (-\ 6/8 \log2(6/8) - 2/8 \log2(2/8)) =$$

0.892 bits.

ويكون عائد المعلومات:

$$Gain(x2) = 0.940 - 0.892 = 0.048 \text{ bits.}$$

بالمقارنة بين معدلي العائدَين للاختبارين x1 و x2 فإن الخوارزمية ستختار الاختبار الأول ليكون اختبار البدء في تقسيم T و بناء الشجرة لأن عائده أكبر.

إحدى طرق معالجة القيم المستمرة:

1. يتم أولاً فرز أمثلة التدريب اعتماداً على قيم الخاصية Y إلى مجموعة مرتبة من القيم بدون تكرار ولتكن {v1, …, vm}.
2. أي قيمة عتبة مثلى تقع بين vi و vi+1 ، ستؤدي إلى تقسيم الأمثلة اعتماداً على قيمة الخاصية Y إلى مجالين: مجال أصغر أو يساوي من Y {v1, …, vi} ومجال اكبر من Y {vi+1, …, vm}
3. عادةً ما يتم اختيار قيمة العتبة لتكون نقطة المنتصف للفترة (vi + vi+1) / 2. لضمان ان تكون قيمة العتبة من نفس قيم قاعدة البيانات أو متوسط حسابي أو يتم تحديدخا من خلال خوارزميات إيجاد حل أمثلي أخرى.

1. مجموعة قيم العتبة المثالية Z (اعتماداً على C4.5) هي {65, 70, 75, 78, 80, 85, 90, 95}.
2. هنا يجب اختيار قيمة للعتبة Z تكون مثالية من هذه المجموعة، ومن أجل المثال تكون (المتوسط أو الربح الأكبر) Z = 80 و الاختبار x3 هو:

Attribute2 ≤ 80 or Attribute2 > 80.

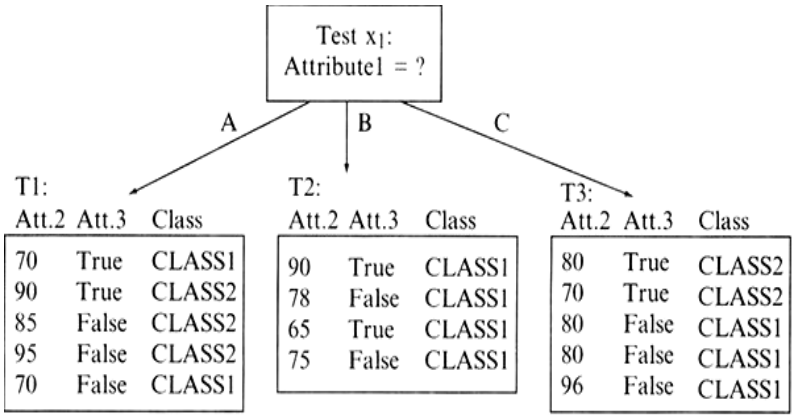
وعليه:

$$Infox3(T) = 9/14\ (-\ 7/9\ \log2(7/9) - 2/9\ \log2(2/9))$$
$$+\ 5/14\ (-\ 2/5\ \log2(2/5) - 3/5\ \log2(3/5)) = 0.837\ bits.$$

وبالتالي يكون الربح :

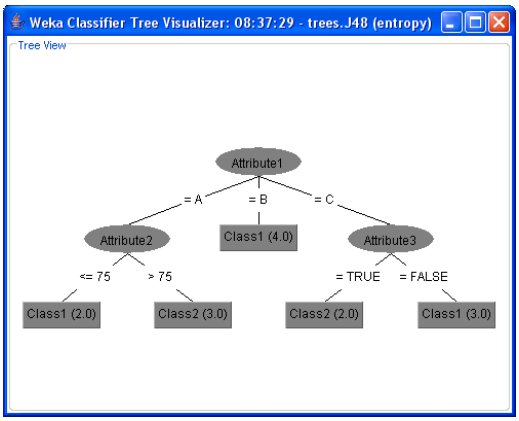
$$Gain(x3) = 0.940 - 0.837 = 0.103\ bits.$$

إذا ما قارنّا عائد المعلومات للخواص الثلاثة، سنجد أن الخاصية الأولى ما تزال تقدم أكبر عائد بقيمة 0.246 بت. وبالتالي سيتم اعتمادها لتكون خاصية البداية لإنشاء شجرة القرار. كما في الشكل التالي.



وبما أن العقدة الابن الناتجة عن الاختبار x1 حيث Attribute1 = B لها أربع حالات (أمثلة) تنتمي جميعها إلى الصف الأول، فإن هذه العقدة ستكون ورقة ويمكن استثناؤها من باقي الاختبارات.

وبتطبيق نفس الخطوات على الخاصيتين الباقيتين بأسلوب عودي تنتج شجرة قرار نهائية تتضح في الشكل التالي:



يمكن تمثيل شجرة القرار بطريقة مختلفة، أي باستخدام كود تنفيذي باستخدام قواعد If – then الشرطية لتفريع بنية الشجرة.

وتصبح مجموعة القواعد كما يلي:

Rule1: If Attribute1=A and Attribute2<=75 Then Classification=Class1;

Rule2: If Attribute1=A and Attribute2>75 Then Classification=Class2;

Rule3: If Attribute1=B Then Classification=Class1;

Rule4: If Attribute1=C and Attribute3=True Then Classification=Class2;

Rule5: If Attribute1=C and Attribute3=False Then Classification=Class1;

تطبيق خوارزمية c4.5 باستخدام برمجية ويكا

1) مقدمة: Introduction

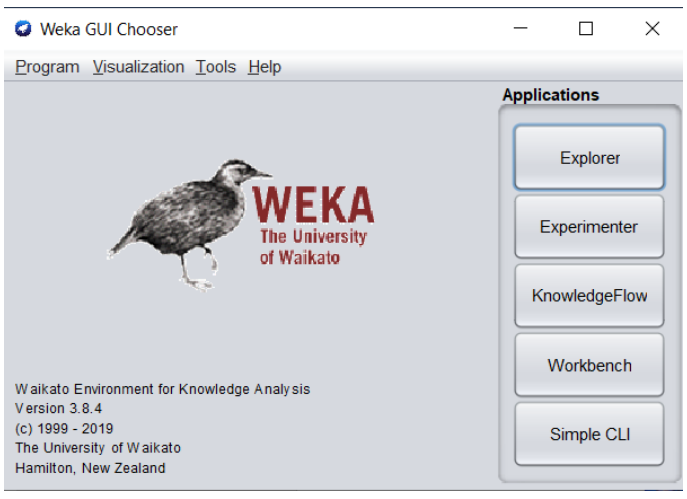
برنامج Weka هو برنامج تعلم ذاتي مكتوب بلغة جافا، ويحوي العديد من خوارزميات التعلم الذاتي. وهو مكتوب من قبل جامعة Waikato في نيوزيلاند.

2) مزايا البرنامج Weka: Weka Features

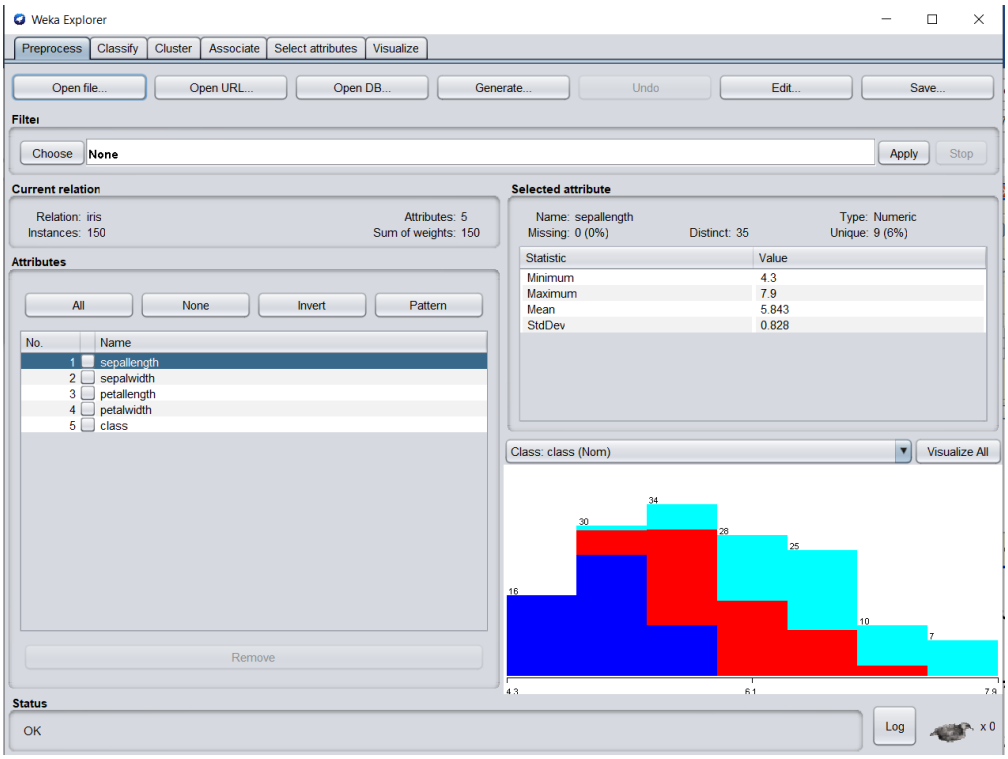
- برنامج خاص بخوارزميات التنقيب عن البيانات وتعلم الآلة، مكتوب بلغة الجافا، ومتاح للاستخدام المجاني GNU.
- يستخدم من أجل البحث، والتعليم، والتطبيقات.
- مجموعة كاملة من أدوات المعالجة المسبقة للبيانات.
- واجهة مستخدم رسومية.

3) المستكشف: Explorer

عندما تشغل البرنامج ، تظهر الواجهة الرئيسية له كما هو مبين في الشكل التالي.



يؤدي النقر فوق الزر Explorer (القسم الذي تم التعامل معه في هذا الدرس) إلى فتح مستكشف Weka الذي يتم فيه تجهيز البيانات، وتطبيق خوارزميات التعلم المطلوبة، ومقارنة النتائج. يبين الشكل التالي مستكشف Weka.



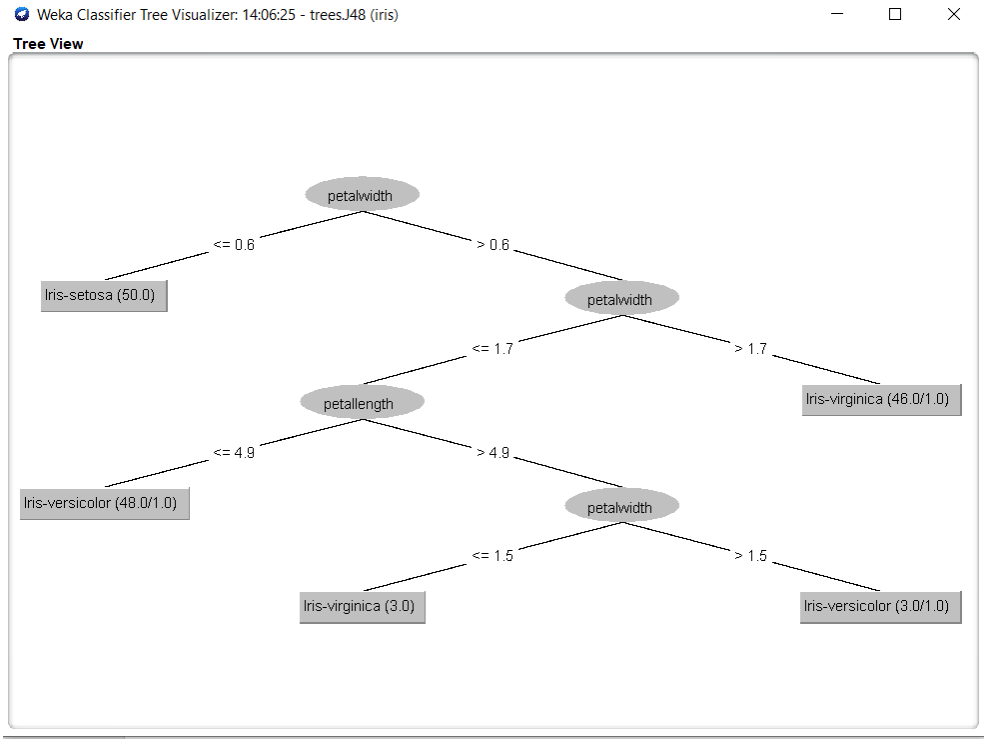
يحوي المستكشف المزايا التالية فيما يتعلق بالمعالجة المسبقة للبيانات:

- يمكن استيراد البيانات من ملف بتنسيقات مختلفة مثل: ARFF, CSV, C4.5, binary.
- يمكن أيضاً قراءة البيانات من عنوان انترنت URL أو من قاعدة بيانات SQL (باستخدام JDBC).
- تدعى أدوات المعالجة المسبقة في Weka بالفلاتر filters، حيث يحوي البرنامج فلاتر للتقطيع Discretization، والتوحيد normalization، وأخذ العينات resampling، واختيار الخصائص attribute selection، وغير ذلك من الفلاتر.

4) تطبيق خوارزمية C4.5 باستخدام Weka:

يتم ذلك بإتباع الخطوات التالية:

1. حفظ ملف مجموعة التدريب بحيث يحوي ورقة عمل واحدة ضمن البرنامج Excel بامتداد csv (تفصل بين القيم فاصلة عادية,).
2. فتح برنامج Weka في وضعية المستكشف Explorer.
3. التوجه إلى علامة التبويب Preprocess ثم النقر فوق الزر open file واختيار الملف السابق ذي الامتداد csv. واختيار جميع الخصائص.
4. التوجه إلى علامة التبويب Classify وانتقاء classifier-trees-j48(C4.5).
5. يمكن اختيار مجموعة الاختبار، ثم نقر زر Start، فتظهر النتيجة بالتفصيل.
6. انقر بالزر الأيمن فوق عنوان النتيجة في القسم الأيسر وانتق visual tree.



5) تطبيق عملي2:

تطبيق مجموعة البيانات IRIS التي تصنف نوع أزهار إلى ثلاثة أصناف وفق 4 خصائص وفق ما يلي:

- عدد الأمثلة: 150 (50 مثال لكل صنف)
- عدد الخصائص: 4 مستمرة
- عدد الأصناف:3

أما الخصائص فهي:

- طول كأس الزهرة Sepal length ومجال قيمه: 4.3-7.9
- عرض كأس الزهرة Sepal width ومجال قيمه: 2-4.4
- طول البتلة Petal length ومجال قيمه: 1-6.9

- عرض البتلة Petal width ومجال قيمه: 0.1-2.5

علماً أن ملف مجموعة البيانات مرفق مع البرنامج weak ضمن المجلد data\iris.arff

6) ملحق الجلسة الرابعة: خوارزمية C4.5 باستخدام بايثون

1. مكتبات بايثون المستخدمة:

NumPy: is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
Original author: Travis Oliphant
Initial release: As Numeric, 1995; as NumPy, 2006
Written in: Python, C

Pandas: is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.
Original author: Wes McKinney
Initial release date: 11 January 2008
Written in: Python, Cython, C

sklearn : is a software machine learning library for the Python programming language. It features various classification, regression and clustering, and it is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.
Original author: David Cournapeau
Initial release date: June 2007
Written in: Python, Cython, C, C++

Matplotlib: is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits.
Original author: John D. Hunter
Initial release: 2003
Written in: Python

2. مراحل تطبيق الخوارزمية:

بفرض لدينا قاعدة البيانات Drugs.csv طبق باستخدام لغة البرمجة بايثون خوارزمية c4.5 وارسم شجرة القرار.

1) **تحميل البيانات(Loading Data)**

```
In [1]: import numpy as np
        import pandas as pd
        from sklearn.tree import DecisionTreeClassifier
```

```
In [2]: my_data = pd.read_csv("drug200.csv", delimiter=",")
        my_data[0:5]
```

Out[2]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY

2) معالجة البيانات واختيار الخصائص (Preprocessing Data and Feature Selection)

```
In [3]: X = my_data[['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K']].values
        X[0:5]
```

```
Out[3]: array([[23, 'F', 'HIGH', 'HIGH', 25.355],
               [47, 'M', 'LOW', 'HIGH', 13.093],
               [47, 'M', 'LOW', 'HIGH', 10.113999999999999],
               [28, 'F', 'NORMAL', 'HIGH', 7.797999999999999],
               [61, 'F', 'LOW', 'HIGH', 18.043]], dtype=object)
```

```
In [4]: from sklearn import preprocessing
        le_sex = preprocessing.LabelEncoder()
        le_sex.fit(['F','M'])
        X[:,1] = le_sex .transform(X[:,1])
        le_BP = preprocessing.LabelEncoder()
        le_BP.fit([ 'LOW', 'NORMAL', 'HIGH'])
        X[:,2] = le_BP.transform(X[:,2])
        le_Chol = preprocessing.LabelEncoder()
        le_Chol.fit([ 'NORMAL', 'HIGH'])
        X[:,3] = le_Chol.transform(X[:,3])
        X[0:5]
```

```
Out[4]: array([[23, 0, 0, 0, 25.355],
               [47, 1, 1, 0, 13.093],
               [47, 1, 1, 0, 10.113999999999999],
               [28, 0, 2, 0, 7.797999999999999],
               [61, 0, 1, 0, 18.043]], dtype=object)
```

```
In [5]: y = my_data["Drug"]
        y[0:5]
```

```
Out[5]: 0    drugY
        1    drugC
        2    drugC
        3    drugX
        4    drugY
        Name: Drug, dtype: object
```

3) تقسيم البيانات (Splitting Data)

```
In [6]: from sklearn.model_selection import train_test_split

In [7]: X_trainset, X_testset, y_trainset, y_testset = train_test_split(X, y, test_size=0.3, random_state=3)
```

4) بناء نموذج المصنف وتدريبه واختباره (Building/ training/ testing (Decision Tree Model

```
In [8]: drugTree = DecisionTreeClassifier(criterion="entropy", max_depth = 4)
        drugTree # it shows the default parameters

Out[8]: DecisionTreeClassifier(criterion='entropy', max_depth=4)

In [9]: clf =drugTree.fit(X_trainset,y_trainset)

In [10]: clfp = drugTree.predict(X_testset)

In [11]: print (clfp [0:5])
         print (y_testset [0:5])

['drugY' 'drugX' 'drugX' 'drugX' 'drugX']
40     drugY
51     drugX
139    drugX
197    drugX
170    drugX
Name: Drug, dtype: object
```

5) تقييم النموذج (Evaluating Model)

```
In [12]: from sklearn import metrics
         import matplotlib.pyplot as plt
         print("DecisionTrees's Accuracy: ", metrics.accuracy_score(y_testset, clfp))

DecisionTrees's Accuracy:  0.9833333333333333
```

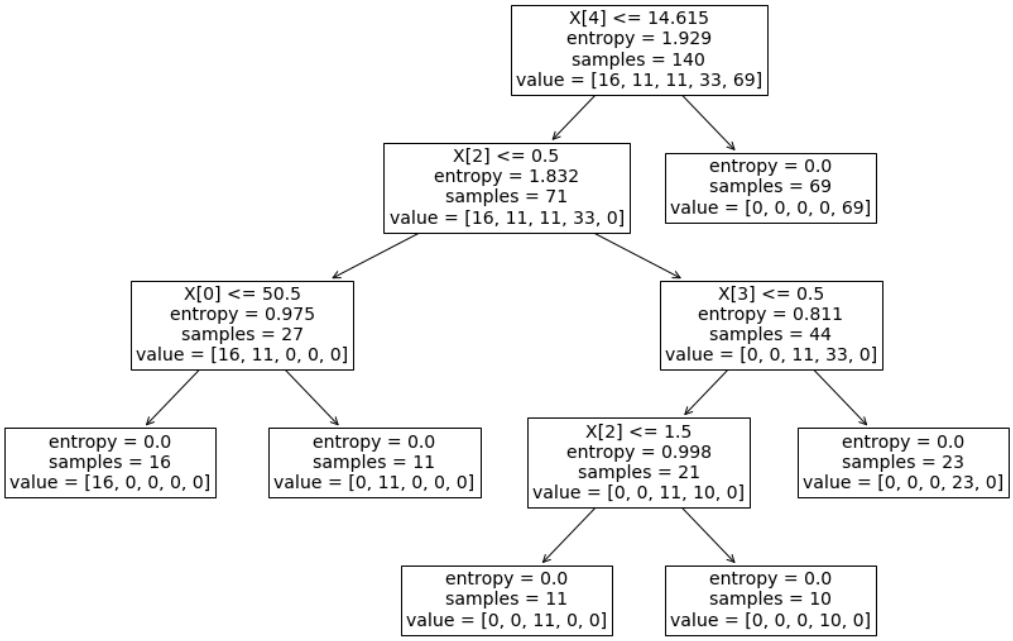
6) رسم شجرة القرار(Visualizing Decision Trees)

```
In [13]: import matplotlib.pyplot as plt

         plt.figure(figsize=(15,10))

         tree.plot_tree(clf, fontsize=14)  # doctest: +SKIP

Out[13]: [Text(523.125, 489.24, 'X[4] <= 14.615\nentropy = 1.929\nsamples = 140\nvalue = [16, 11, 11, 33, 69]'),
          Text(418.5, 380.52000000000004, 'X[2] <= 0.5\nentropy = 1.832\nsamples = 71\nvalue = [16, 11, 11, 33, 0]'),
          Text(209.25, 271.8, 'X[0] <= 50.5\nentropy = 0.975\nsamples = 27\nvalue = [16, 11, 0, 0, 0]'),
          Text(104.625, 163.08000000000004, 'entropy = 0.0\nsamples = 16\nvalue = [16, 0, 0, 0, 0]'),
          Text(313.875, 163.08000000000004, 'entropy = 0.0\nsamples = 11\nvalue = [0, 11, 0, 0, 0]'),
          Text(627.75, 271.8, 'X[3] <= 0.5\nentropy = 0.811\nsamples = 44\nvalue = [0, 0, 11, 33, 0]'),
          Text(523.125, 163.08000000000004, 'X[2] <= 1.5\nentropy = 0.998\nsamples = 21\nvalue = [0, 0, 11, 10, 0]'),
          Text(418.5, 54.360000000000014, 'entropy = 0.0\nsamples = 11\nvalue = [0, 0, 11, 0, 0]'),
          Text(627.75, 54.360000000000014, 'entropy = 0.0\nsamples = 10\nvalue = [0, 0, 0, 10, 0]'),
          Text(732.375, 163.08000000000004, 'entropy = 0.0\nsamples = 23\nvalue = [0, 0, 0, 23, 0]'),
          Text(627.75, 380.52000000000004, 'entropy = 0.0\nsamples = 69\nvalue = [0, 0, 0, 0, 69]')]
```



3. التطبيق العملي: لعب الغولف

يطلب حساب وبناء شجرة القرار يدوياً وباستخدام Weka ولة البرمجة بايثون لمجموعة البيانات التالية:

Outlook	Temperature	Humidity	Windy	Play (positive) / Don't Play (negative)
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

1. بفرض وجود البيانات التالية المتعلقة بلعب الغولف حسب الطقس، وبتطبيق خوارزمية C4.5 يطلب ما يلي:

outlook	humidity	windy	play
sunny	high	FALSE	no
sunny	high	TRUE	no
overcast	high	FALSE	yes
overcast	normal	TRUE	yes
sunny	high	FALSE	no
sunny	normal	FALSE	yes
sunny	normal	TRUE	yes
overcast	high	TRUE	yes

a. ارسم الشجرة الناتجة بدون حسابات على افتراض أن الخاصية الأولى للتفريع هي outlook.

b. صنف المثال التالي حسب الشجرة السابقة:

Outlook= overcast and humidity=normal and windy=FALSE: play=?

4. الوظيفة: تصنيف الأزهار باستخدام لغة البرمجة بايثون والبرمجية ويكا

طبق الخوارزمية على قاعدة البيانات iris من أجل نسبة تقسيم 60% باستخدام لغة البرمجة بايثون خوارزمية c4.5 وارسم شجرة القرار.
ملاحظة يمكنك تحميل بيانات قاعدة البيانات iris باستخدام التعليمات التالية:

```
In [16]: from sklearn.datasets import load_iris
         from sklearn import tree
         X, y = load_iris(return_X_y=True)
```

د. سوسن اسجيع