وزارة التعليم العالي

جامعة حمص

الكلية التطبيقية

قسم تقنيات حاسوب

السنة الثالثة

قواعد معطيات٣

القسم العملي

المحاضرة السادسة

اعداد المدرسين:

م. زينب مراد **م. بتول الليوس**

Classification in Datamining

What is classification ?

- Classification is a supervised learning task.
  - Supervision, the input data are associated with labels indicating the class of each tuple (labeled dataset).
- The process of finding a model (or function) that describes and distinguishes data classes or concepts.

التصنيف في التنقيب عن البيانات هو عملية إيجاد نموذج (أو دالة) يصف ويميز بين فئات أو مفاهيم البيانات المختلفة

– يُستخدم التصنيف كتقنية تعلم آلي مراقب لتوقع تصنيف أو فئة عنصر بيانات جديد بناءً على مجموعة بيانات تدريبية تحتوي على أمثلة معروفة الفئات.

– يتكون الإدخال من سجلات بيانات(tuples) تتميز بمجموعة من الخصائص (السمات) مع تسمية فئة مرتبطة بها.

– الهدف هو تعلم نموذج تصنيف(Classifier) يمكنه تعيين الفئة الصحيحة للبيانات الجديدة غير المعروفة.

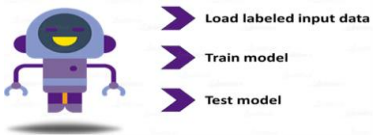
- Usually, the given data set is divided into **training and test sets**

  (i.e., training & test datasets must have the same distribution).

- **A General Approach For Building Classification Models (Two-step Process):**

  o learning step or training phase.

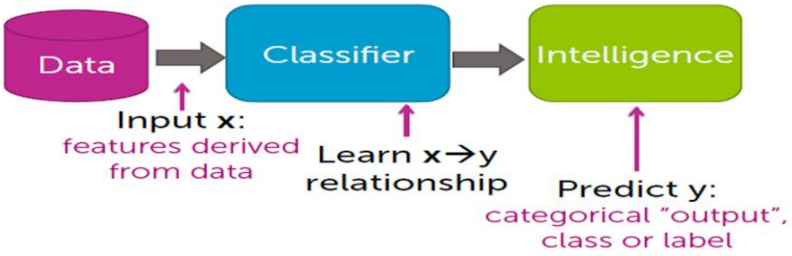
  o Testing Phase.



- **Model usage:** To classify **"future", "unknown" or "previously unseen"** data tuples

  (for which the class label is not known).

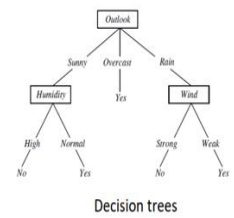
- Given a collection of data tuples (training set). Each tuple is characterized by (x, y):
  - ✓ x (input) is the ***attribute set***.
  - ✓ y (output) is the ***class label attribute***.
- Task: based on the input dataset, learn a ***classification model*** (***Classifier***) that describes a predetermined set of data classes.



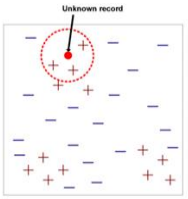
تشمل خوارزميات التصنيف الشائعة شجرة القرار، الانحدار اللوجستي، نايف بايز، آلات الدعم الناقل، والشبكات العصبية.

يستخدم التصنيف في مجالات متعددة مثل التشخيص الطبي، كشف البريد المزعج، تقسيم العملاء، والتنبؤ بسلوك المستخدم

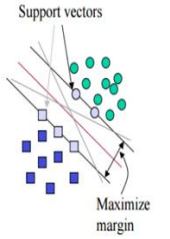
- Base Classifiers
  - ✓Decision trees
  - ✓Bayesian classifiers
  - ✓Support vector machines
  - ✓Nearest neighbors
  - ✓Rule-based Methods
  - ✓Neural Networks
  - ✓.....
- Ensemble Classifiers
  - ✓Boosting,
  - ✓Bagging,
  - ✓Random Forests.



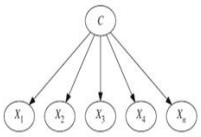
Decision trees



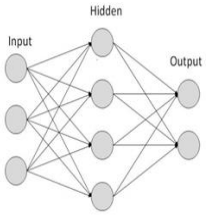
k nearest neighbours



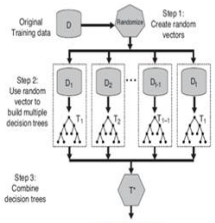
Support vector machines



Bayesian classifiers



Neural networks



Ensembles

🌳 Decision Tree: شجرة القرار

➢ A decision tree is a flowchart-like tree structure where:

✓ Each internal node denotes a test on an attribute.

✓ Each branch represents an outcome of the test.

✓ And each leaf node holds a class label, the topmost node in the tree is the root node.

➢ Given a tuple, X, for which the class label is unknown,

✓ The attribute values of the tuple X are tested against the decision tree.

✓ Therefore, a path is traced from the root to a leaf node which holds the class prediction for that tuple.

شجرة القرار هي خوارزمية تعلم تحت الإشراف تُستخدم في التصنيف و الانحدار. تعمل مثل مخطط انسيابي يوجّه القرارات خطوة بخطوة.

العقدة الجذرية(Root Node): نقطة البداية وتمثل مجموعة البيانات كاملة

العقد الداخلية(Internal Nodes): كل عقدة تمثل اختبارًا على خاصية (مثل: "هل العمر > 30؟ ")

الفروع(Branches): كل فرع يمثل نتيجة الاختبار (مثل: نعم/لا)

العقد الورقية(Leaf Nodes) : العقد النهائية التي تعطي التصنيف أو القيمة المتوقعة

Decision Tree Construction:

➢ A greedy algorithm, where the tree is constructed in a top-down recursive approach.

➢ At the start, **all the training tuples are at the root node.**

Then **training set is recursively partitioned into smaller subsets as the tree is being built.**

➢ **training tuples are partitioned based on test attributes which are selected based on**

📊 **A heuristic or statistical measure**, like:

✓ **Information Gain.**

✓ **Gain Ratio.**

شجرة القرار تُبنى باستخدام خوارزمية جشعة (Greedy Algorithm) أي انها تختار في كل خطوة افضل تقسيم ممكن دون النظر الى المستقبل

* خطوات البناء:

1 - (Root Node) البداية من الجذر جميع بيانات التدريب (Training Tuples)تكون في العقدة الجذرية

2-(Recursive Partitioning) التقسيم التكراري
يتم تقسيم مجموعة البيانات تدريجيًا إلى مجموعات أصغر مع بناء الشجرة خطوة بخطوة.

3-(Attribute Selection) اختيار الخاصية
يتم اختيار الخاصية التي تحقق أفضل تقسيم للبيانات اعتمادًا على مقياس إحصائي أوخوارزمي.

📊 المقاييس المستخدمة لاختيار الخاصية:

✓ Information Gain (مكسب المعلومات)
يقيس مقدار تقليل عدم اليقين عند تقسيم البيانات باستخدام خاصية معينة (Entropy)

كلما زاد مكسب المعلومات، كانت الخاصية أفضل للتقسيم

✓ Gain Ratio (نسبة المكسب)
تعديل لمكسب المعلومات يأخذ في الاعتبار عدد الفروع الناتجة عن التقسيم، لتجنب الانحياز نحو الخصائص ذات القيم الكثيرة

Heart Disease Prediction:

We'll try to learn a classifier capable of predicting whether or not someone has heart disease or not. Our dataset consists of the following attributes:

#	Attribute	Explanation
1	Age	Age of the patient.
2	Sex	Symmetric binary attribute, 1 for male and 0 for female.
3	cp	Chest Pain type (1 to 4).
4	trestbps	resting blood pressure in (mm Hg).
5	chol	serum cholesterol in mg/dl.
6	fbs	(fasting blood sugar > 120 mg/dl) (1 = true; 0 = false).
7	thalach	maximum heart rate achieved.
8	**Target (class label attribute)**	**Asymmetric binary attribute. 1 disease, 0 no disease**

You can see more details about this data using this link (https://archive.ics.uci.edu/ml/datasets/heart+disease).

1- Import required libraries and load dataset: استيراد المكتبات

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score          #the function is used for evaluating the learned classifier
from sklearn import tree import graphviz            #for tree visualization.
```

2- load our dataset: قراءة او تحميل البيانات من كامل قاعدة البيانات

```
heart_disease_df = pd.read_csv('heart-disease.csv')
```

اظهار اول خمس اسطر او معلومات عن قاعدة البيانات:

```
heart_disease_df.head()
```

Out[2]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
heart_disease_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

3- Explore Target (class label attribute):

 # getting information about our target attribute:

```
heart_disease_df['target'].value_counts()
```

```
Out[4]: 1    165
        0    138
        Name: target, dtype: int64
```

4- Explore some attributes relationship with target attribute:

we'll combine a couple of attributes such as (age, thalach) and compare them to our target using scatter plot :

```
# create a figure :          # انشاء المخطط وتحديد حجمه
plt.figure(figsize=(10,6))
# draw a scatter plot :      # تعليمات رسم المخطط
plt.scatter(heart_disease_df.age[heart_disease_df.target==1],heart_disease_df.thalach[h eart_disease_df.target==1],c='red')
plt.scatter(heart_disease_df.age[heart_disease_df.target==0],heart_disease_df.thalach[h eart_disease_df.target==0],c='blue')
plt.title('Heart Disease in function of Age and Max Heart Rate')
plt.xlabel('Age')
plt.ylabel('Max Heart Rate')
plt.legend(["disease" , 'no disease'])
```

3- Prepare our Dataset for classification:

In this step, we're going split our dataset into training and test sets:

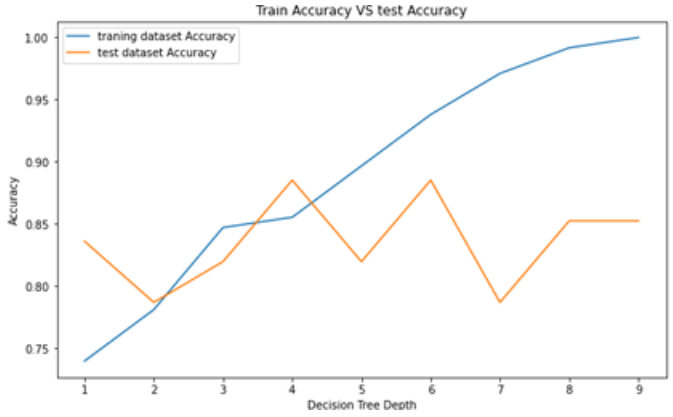
```
# apply classification using our given attributes :      # فصل الخصائص عن الهدف
x = heart_disease_df.drop(["target"],axis=1)             # الخصائص
# split our target from our DF:
y = heart_disease_df['target']                           # الهدف (١ = مريض قلب, ٠ = سليم)
# split our data into training and test sets:
np.random.seed(42)
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size= .2)      # تقسيم البيانات
```

5- Build Decision Tree Model:

We're going to use DecisionTreeClassifier class provided by sklearn in order to learn a model based on the input data.

Our class takes some important parameters such as criterion and max_depth.

```
train_acc = []                #training set accuracy list
 test_acc = []                #test set accuracy list
train_error = []              #training set error list
test_error= []                #test set error list
depths = list(range(1,10))
for i in range (1,10):
    dt = DecisionTreeClassifier(criterion='entropy',max_depth=i)
    dt.fit(x_train,y_train)
train_acc.append(accuracy_score(y_train,dt.predict(x_train)))
test_acc.append(accuracy_score(y_test,dt.predict(x_test)))
train_error.append(1-(accuracy_score(y_train,dt.predict(x_train))))
test_error.append(1-(accuracy_score(y_test,dt.predict(x_test))))
plt.figure(figsize=(10,6))
plt.plot(depths, train_acc,'-',depths,test_acc,'-')
plt.title("Train Accuracy VS test Accuracy")
plt.xlabel("Decision Tree Depth")
plt.ylabel("Accuracy")
plt.legend(["traning dataset Accuracy","test dataset Accuracy"])
```



✓ We can notice that if we use the training set to measure the classifier's accuracy, this estimate would likely be optimistic because the classifier tends to overfit the data.

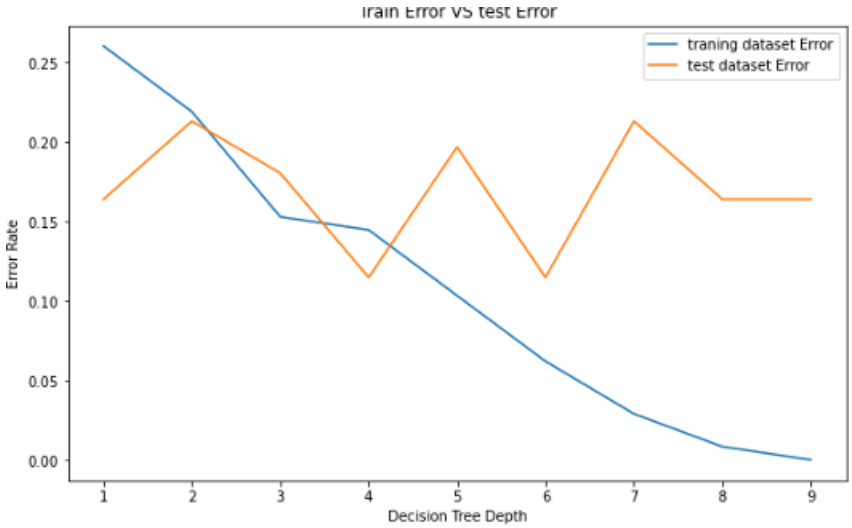
✓ Because of that, we tend to estimate classifier accuracy using a list of tuples called test dataset which is independent of the training dataset.

✓ The accuracy of a classifier, M, on a given test set is the "percentage of test set tuples that are correctly classified by the classifier".

✓ We can evaluate the same concepts based on classification error which is given by:

- Error = 1- Accuracy.

```
plt.figure(figsize=(10,6))
plt.plot(depths, train_error,'-',depths,test_error,'-')
plt.title("Train Error VS test Error")
plt.xlabel("Decision Tree Depth")
plt.ylabel("Error Rate")
plt.legend(["traning dataset Error","test dataset Error"])
```



6- Select DecisionTreeClassifier max_depth Parameter:

✓ To select the best max_depth value for a decision tree, we should notice the effect of getting higher depth trees VS lower depth trees.

✓ Now, we're going to build a new DecisionTreeClassifier based only on two attributes (age, thalach), in order to notice the max_depth parameter effect for 2 values (9 and 4).

```
#extract (age, thalach) attributes, y is same as above.
x1 = heart_disease_df[["age","thalach"]] x1_train, x1_test, y1_train, y1_test =
train_test_split(x1,y,test_size= .2) #initialize the tree classifier and fit the model dt1 =
DecisionTreeClassifier(criterion='entropy',max_depth=9)
 #try with 4 also
dt1.fit(x1_train,y1_train)
```

```
#draw the fitted model:

def visualize_classifier(model, X, y, ax=None, cmap='rainbow'):

 ax = ax or plt.gca()        #create figure
```

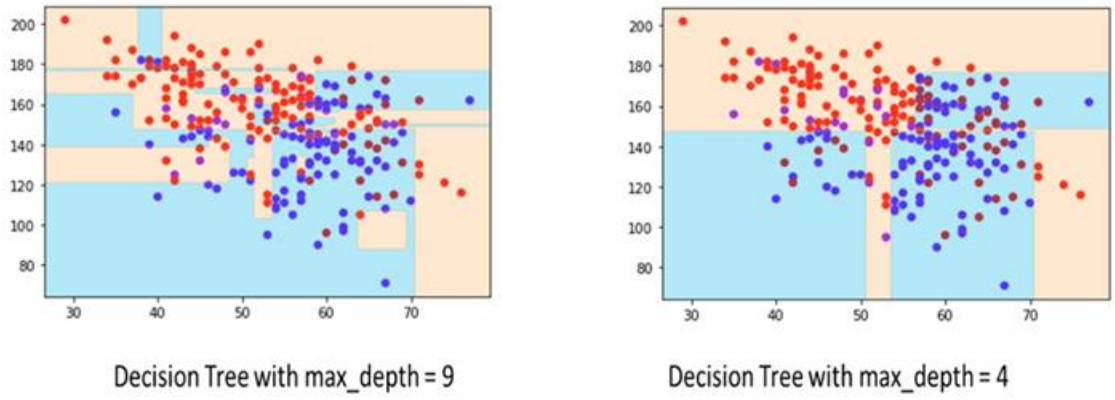
 # Plot the training points

```
ax.scatter(X["age"], X["thalach"], c=y, s=30, cmap=cmap)

xlim = ax.get_xlim()

 ylim = ax.get_ylim()

 xx, yy = np.meshgrid(np.linspace(*xlim, num=200),np.linspace(*ylim, num=200))

Z = model.predict(np.c_[xx.ravel(), yy.ravel()]).reshape(xx.shape)
```

 # Create a color plot with the results

```
n_classes = len(np.unique(y))

contours = ax.contourf(xx, yy, Z, alpha=0.3, levels=np.arange(n_classes + 1) - 0.5, cmap=cmap)
ax.set(xlim=xlim, ylim=ylim) visualize_classifier(dt1, x1_train, y1_train)
```

the figure on the left when max_depth = 9 and the other is when max_depth =4:



This where tree pruning is an important task to work with.

➢ When a decision tree is built, many of the branches will reflect anomalies in the training data due to noise or outliers.

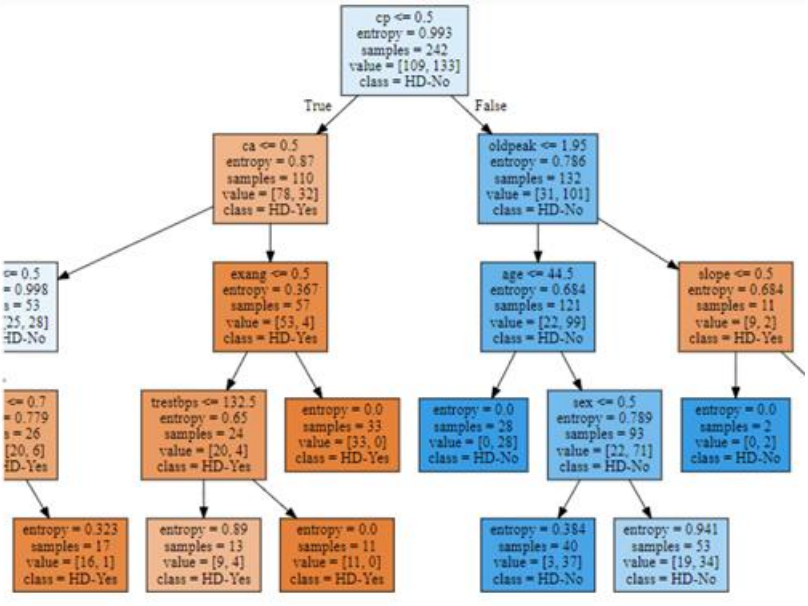
➢ Pruned trees:

✓ Tend to be smaller and less complex, and thus easier to comprehend.

✓ Usually faster and better at classifying test tuples than unpruned trees.

7- Optional Section visualize Decision Tree:

```
dt = DecisionTreeClassifier(criterion='entropy',max_depth=4)
dt.fit(x_train,y_train)
dot_data=tree.export_graphviz(dt,out_file=None,feature_names=list(x.columns),
class_names=['HD-Yes', 'HD-No'], filled = True)

tr = graphviz.Source(dot_data, format ="png")
```



References:

1- Pandas docs [https://pandas.pydata.org/docs/].
2- NumPy docs [https://numpy.org/doc/].
3- Matplotlib docs [https://matplotlib.org/3.3.3/contents.html].
4- SciKit-learn docs [https://scikit-learn.org/0.21/documentation.html].