# PRACTICA 23

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
drugs = pd.read_csv('C:/Users/Isaac/Desktop/IHD/EBAC DT/M23 DS/drugs.csv')
drugs
```

Out[3]:

| | 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 |

In [4]:
```
# defininimos caracteristicas para los pronosticos

feature_cols = ['Age','Sex','BP','Cholesterol','Na_to_K']

# definimos valores para 'x' y 'y'

X = drugs[feature_cols].values
y = drugs.Drug
```

Convertir las variables predictoras cualitativas de esta base a una escala numérica mediante la instrucción "preprocessing.LabelEncoder()

In [5]:
```
from sklearn import preprocessing

cod_sex = preprocessing.LabelEncoder()
cod_sex.fit(['F','M'])
X[:,1] = cod_sex.transform(X[:,1])

cod_bp = preprocessing.LabelEncoder()
cod_bp.fit(['HIGH','LOW','NORMAL'])
X[:,2] = cod_bp.transform(X[:,2])

cod_Chol = preprocessing.LabelEncoder()
cod_Chol.fit(['HIGH','NORMAL'])
X[:,3] = cod_Chol.transform(X[:,3])
```

In [6]:
```
# creamos grupos de entrenamiento y prueba

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 1)
```

In [7]:
```
# objeto de calsificacion del Arbol de Decision

clf = DecisionTreeClassifier()

# entrenamos el modelo

clf.fit(X_train, y_train)
```

Out[7]:
```
DecisionTreeClassifier
DecisionTreeClassifier()
```

In [8]:
```
# realizamos predicciones

clf_pred = clf.predict(X_test)
```

In [10]:
```
# Matriz de Confucion

from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, clf_pred)
cm
```

Out[10]:
```
array([[ 4,  0,  0,  0,  0],
       [ 2,  4,  0,  0,  0],
       [ 0,  0,  4,  0,  0],
       [ 0,  0,  0, 19,  0],
       [ 0,  0,  0,  0, 27]], dtype=int64)
```

ESTADISTICAS DE DESEMPEÑO

In [11]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, clf_pred))
```

```
              precision    recall  f1-score   support

       drugA       0.67      1.00      0.80         4
       drugB       1.00      0.67      0.80         6
       drugC       1.00      1.00      1.00         4
       drugX       1.00      1.00      1.00        19
       drugY       1.00      1.00      1.00        27

    accuracy                           0.97        60
   macro avg       0.93      0.93      0.92        60
weighted avg       0.98      0.97      0.97        60
```

ARBOL DE DECISION (REGLAS)

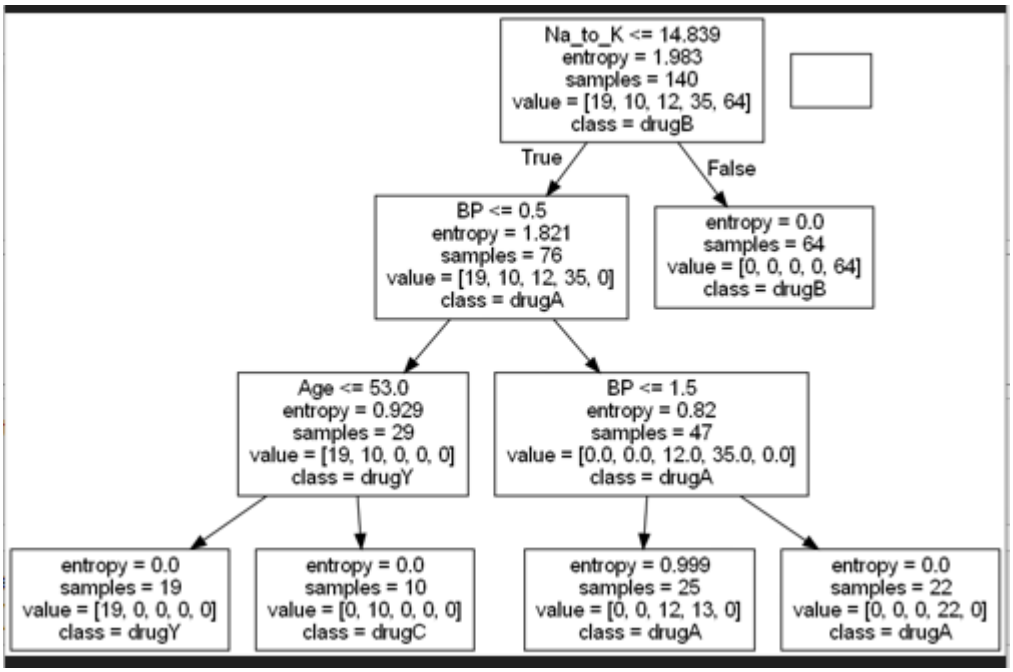
In [13]:
```
dot_data = tree.export_graphviz(clf, out_file = None, feature_names = feature_cols, class_names = ['drugY','drugC','drugX','drugA

# creamos la grafica de Arbol

graph = pydotplus.graph_from_dot_data(dot_data)
```

In [14]:
```
# creacion del Arbol en formato png

graph.write_png('drug.png')
image = Image.open('drug.png')
image.show()
```




## utilizando entropy

In [16]:
```
clf = DecisionTreeClassifier(criterion = 'entropy', max_depth = 3)
clf.fit(X_train, y_train)
clf_pred = clf.predict(X_test)
```

In [18]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, clf_pred))
```

```
              precision    recall  f1-score   support

       drugA       0.67      1.00      0.80         4
       drugB       1.00      0.67      0.80         6
       drugC       0.00      0.00      0.00         4
       drugX       0.83      1.00      0.90        19
       drugY       1.00      1.00      1.00        27

    accuracy                           0.90        60
   macro avg       0.70      0.73      0.70        60
weighted avg       0.86      0.90      0.87        60
```

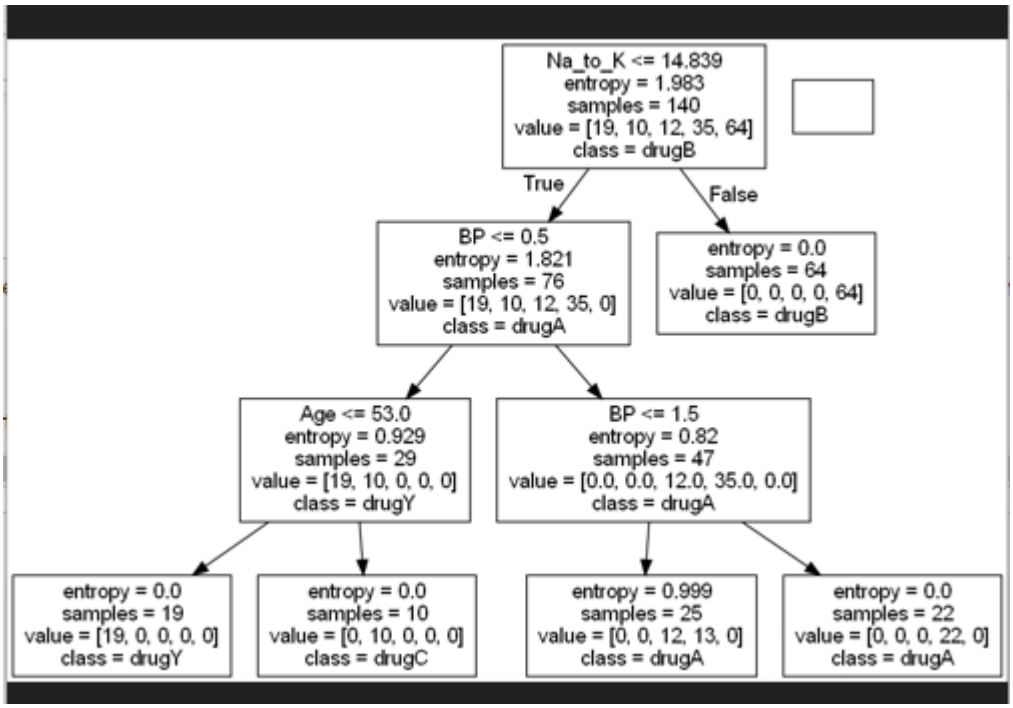
In [19]:
```
dot_data = tree.export_graphviz(clf, out_file = None, feature_names = feature_cols, class_names = ['drugY','drugC','drugX','druga

# creamos la grafica de Arbol

graph = pydotplus.graph_from_dot_data(dot_data)
```

In [27]:
```
# creacion del Arbol en formato png

graph.write_png('Drug.png')
image = Image.open('Drug.png')
image.show()
```




**¿Qué medicamento recomendaría utilizar para un paciente con los siguientes datos?**

| Age | Sex | BP | Cholesterol | Na_to_K |
|---|---|---|---|---|
| 50 | F | HIGH | NORMAL | 15.302 |

In [21]:
```
# Predicción con nuevos valores específicos
new_data = {'Age': 50, 'Sex': 'F', 'BP': 'HIGH', 'Cholesterol': 'NORMAL', 'Na_to_K': 15.302}
new_data['Sex'] = cod_sex.transform([new_data['Sex']])[0]
new_data['BP'] = cod_bp.transform([new_data['BP']])[0]
new_data['Cholesterol'] = cod_Chol.transform([new_data['Cholesterol']])[0]
```

In [26]:
```
new_data_array = np.array([[new_data['Age'], new_data['Sex'], new_data['BP'], new_data['Cholesterol'], new_data['Na_to_K']]])
new_data_array
```

Out[26]:
```
array([[50.   ,  0.   ,  0.   ,  1.   , 15.302]])
```

In [25]:
```
# Realizamos la prediccion
pred_drug = clf.predict(new_data_array)
print("La predicción para el nuevo dato es:", predicted_drug[0])
```

```
La predicción para el nuevo dato es: drugY
```

*** El medicamento que se recomendaria al paciente con los datos dados, sería Drug Y**