

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Minería de Datos

Practica

ARBOL DE

CLASIFICACIÓN

Integrantes:

- **Sánchez Zarazúa Jesua Antonio**

Profesora: Ocampo Botello Fabiola

Grupo: 3CM9

Fecha de entrega: 03/04/2020

Practica. Arboles de Clasificación

INTRODUCCION

Los árboles de decisión son uno de la familia de modelos de aprendizaje automático más utilizados. Se pueden utilizar tanto para resolver problemas de clasificación como de regresión. Una de sus principales ventajas es la facilidad con la que se puede interpretar los resultados en base a reglas. Permitiendo no solo obtener un resultado, sino que inspeccionar los motivos por los que se llega a una predicción dada. Por ejemplo, en un modelo que permita predecir la aparición de fallos en una maquinaria se puede explorar el proceso lógico del algoritmo, identificando los valores de las características que llevar a una conclusión dada. Esto permite no solo utilizar el modelo para predecir un valor, sino actuar sobre las causas para evitar la aparición de resultados no deseados. Siendo la visualización de árboles de decisión una forma de facilitar esto.

Esta practica se realizo usando el lenguaje de programación Python. Además de las librerías para machine learning Scikit Learn y otras librerías específicas como Pandas para la lectura del dataset .CSV y Graphviz para construir una imagen del modelo del árbol resultado del aprendizaje

También se ha de recalcar que se uso Jupyter Notebook como herramienta de desarrollo, por lo que el resultado es un Notebook que posee cada línea de código y resultado obtenido, visible sin la necesidad de compilar el archivo.

INTENCION RECOPIACION DE DATOS

Este conjunto de datos es originalmente del Instituto Nacional de Diabetes y Enfermedades Digestivas y Renales. El objetivo del conjunto de datos es predecir el diagnóstico sobre si un paciente tiene diabetes o no, basándose en ciertas mediciones de diagnóstico incluidas en el conjunto de datos. Se colocaron varias restricciones en el proceso de selección de las instancias que formaban parte de una base de datos más grande. En particular, todos los pacientes aquí son mujeres de al menos 21 años de herencia indígena Pima.

Fuente:

<https://www.kaggle.com/uciml/pima-indians-diabetes-database>

Kaggle Pima Indians Diabetes

Diccionario de Datos


diabetes.csv (23.31 KB)

Views

Pregnancies

Number of times pregnant

Integer




Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	3.85	
Std. Deviation	3.37	
Quantiles		
0	1	3
1	2	5
3	5	7
6	7	9
17	17	100

Glucose

Plasma glucose concentration a 2 hours in an oral glucose tolerance test


Integer



Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	121	
Std. Deviation	32	
Quantiles		
0	99	2
99	117	5
117	141	7
141	199	100

BloodPressure

Integer




Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	120	
Std. Deviation	19	
Quantiles		
0	99	2
99	120	5
120	122	7
122	122	100

SkinThickness

Triceps skin fold thickness (mm)

Integer




Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	20.5	
Std. Deviation	15.9	
Quantiles		
0	0	2
0	23	5
23	32	7
32	99	100

Insulin

2-Hour serum insulin (mu U/ml)

Integer




Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	79.8	
Std. Deviation	115	
Quantiles		
0	0	2
0	32	5
32	32	7
32	32	100


BMI

Body mass index (weight in kg/(height in m)^2)

Decimal



Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	32	
Std. Deviation	7.88	
Quantiles		
0	27.3	2
27.3	32	5
32	36.6	7
36.6	67.1	100




Mean	0.47	
Std. Deviation	0.33	
Quantiles		
0.08	0.08	1
0.24	0.24	2
0.37	0.37	5
0.63	0.63	7
2.42	2.42	100

Age

Age (years)

Integer




Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	33.2	
Std. Deviation	11.8	
Quantiles		
21	24	2
24	29	5
29	41	7
41	81	100

Outcome

Class variable (0 or 1) 268 of 768 are 1, the others are 0

Integer



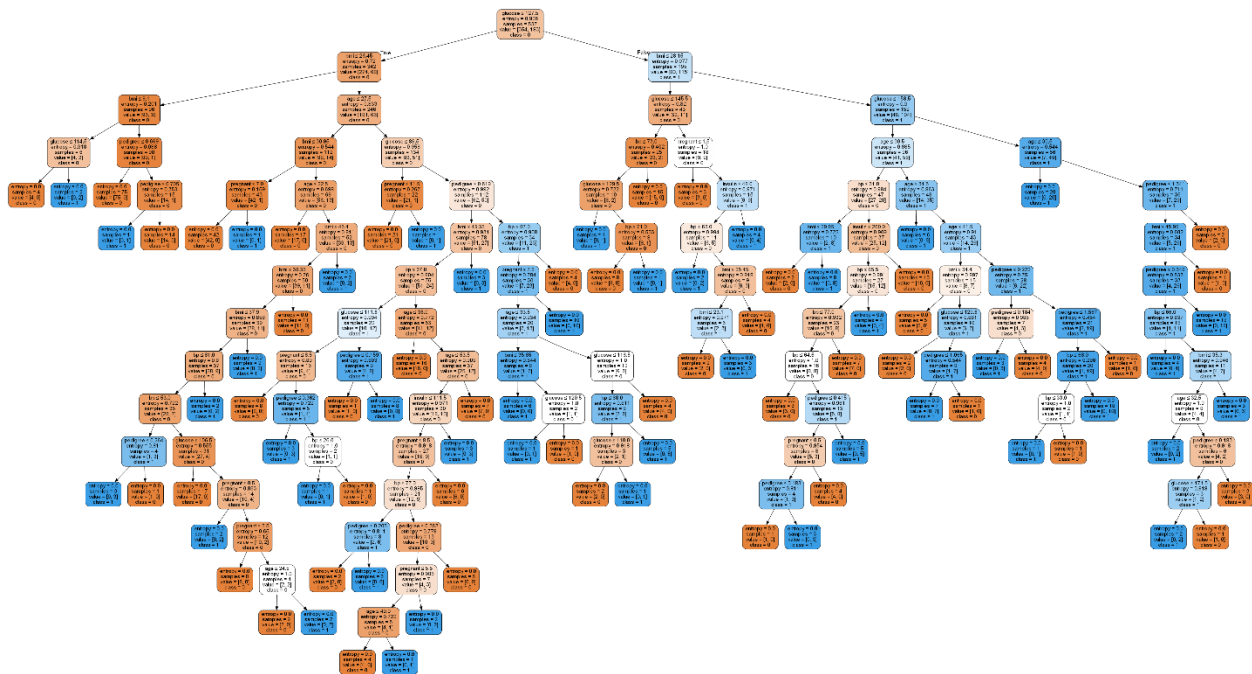
Valid	768	100%
Mismatched	0	0%
Missing	0	0%
Mean	0.35	
Std. Deviation	0.48	
Quantiles		
0	0	2
0	2	5
2	2	7
2	2	100

- Numero de Datos: 768 filas
- Numero de Columnas: 9 Columnas
- Tipos de Datos contenidos: Numericos(int)

OBJETIVO DE ANALISIS DE DATOS

Se entiende que los valores contenidos son indispensables para clasificar casos futuros como pacientes que padecen de diabetes o no, todas las columnas son descritas con tipos de datos numericos y no se encuentran datos no valios dentro del dataset por lo que se puede aceptar el uso de estos en su estado actual para el aprenizaje del arbol, la unica tarea es dividir los datos en dos conjuntos para el entrenamiento y para el prueba.

WORKFLOW ARBOL



MEDIDAS

```
In [20]: #Exactitud  
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Accuracy: 0.7489177489177489

```
In [24]: #Matriz De Confusion  
print(metrics.confusion_matrix(y_test,y_pred))
```

```
[[120  26]  
 [ 32  53]]
```

```
In [27]: #Sensibilidad  
print(metrics.recall_score(y_test,y_pred))
```

0.6235294117647059

```
In [28]: #Precision  
print(metrics.precision_score(y_test,y_pred))
```

0.6708860759493671

```
In [29]: #Este reporte contiene todos los parametros con que dispone la libreria scikit learn para tareas de clasificacion  
print(metrics.classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.79	0.82	0.81	146
1	0.67	0.62	0.65	85
micro avg	0.75	0.75	0.75	231
macro avg	0.73	0.72	0.73	231
weighted avg	0.75	0.75	0.75	231

ANEXO: Cuaderno Completo de Jupyter Notebook

jupyter

ArbolClasificador_DiabetesDataSet

Last Checkpoint: hace 4 minutos (autosaved)

Logout

File

Edit

View

Insert

Cell

Kernel

Help

Trusted

Python 3

In [1]:

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-Learn metrics module for accuracy calculation
from sklearn.tree import export_graphviz
from sklearn.externals.six import StringIO
from IPython.display import Image
import pydotplus
```

In [2]:

```
col_names = ['pregnant', 'glucose', 'bp', 'skin', 'insulin', 'bmi', 'pedigree', 'age', 'label']
# Load dataset
pima = pd.read_csv("diabetes.csv")
pima.columns = col_names
```

In [17]:

```
#split dataset in features and target variable
feature_cols = ['pregnant', 'insulin', 'bmi', 'age', 'glucose', 'bp', 'pedigree']
X = pima[feature_cols] # Features
y = pima.label # Target variable
```

Train Decision Tree Classifier
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)

In [20]:

```
#Exactitud
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

Accuracy: 0.7489177489177489
```

In [24]:

```
#Matriz De Confusion
print(metrics.confusion_matrix(y_test,y_pred))

[[120  26]
 [ 32  53]]
```

In [27]:

```
#Sensibilidad
print(metrics.recall_score(y_test,y_pred))

0.6235294117647059
```

In [20]:


```
print(metrics.classification_report(y_test,y_pred))
```

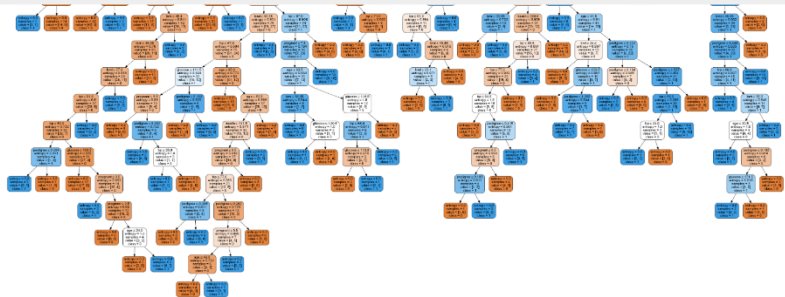
	precision	recall	f1-score	support
0	0.79	0.82	0.81	146
1	0.67	0.62	0.65	85
micro avg	0.75	0.75	0.75	231
macro avg	0.73	0.72	0.73	231
weighted avg	0.75	0.75	0.75	231

In [21]:

```
#Convertimos el modelo del arbol en una imagen que se puede desplegar dentro de Jupyter Notebook
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True, feature_names = feature_cols,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```

Out[21]:





In []: