



Menu ▾

Log in



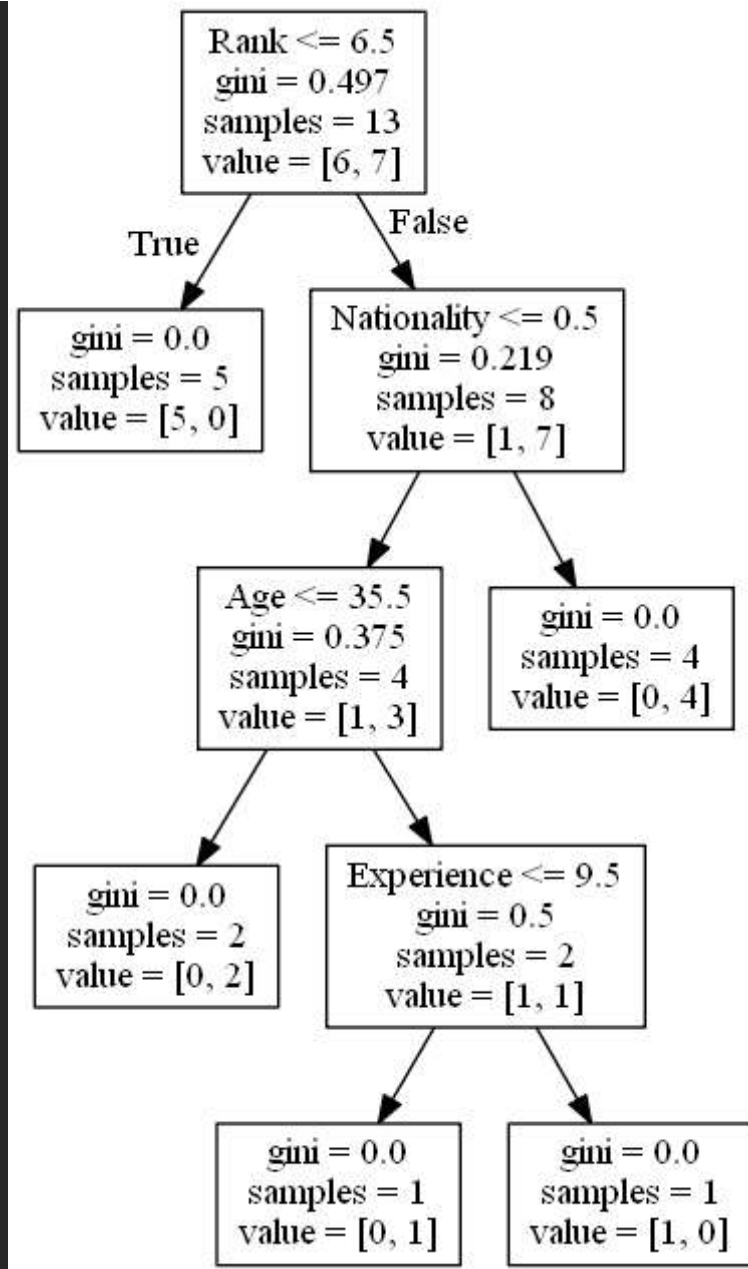
HTML

CSS



Aprendizado de máquina - Árvore de decisão

[< Anterior](#)[Próximo >](#)



Árvore de decisão

Neste capítulo, mostraremos como fazer uma "Árvore de Decisão". Uma árvore de decisão é um fluxograma e pode ajudá-lo a tomar decisões com base na experiência anterior.

No exemplo, uma pessoa tentará decidir se deve ou não ir a um show de comédia.

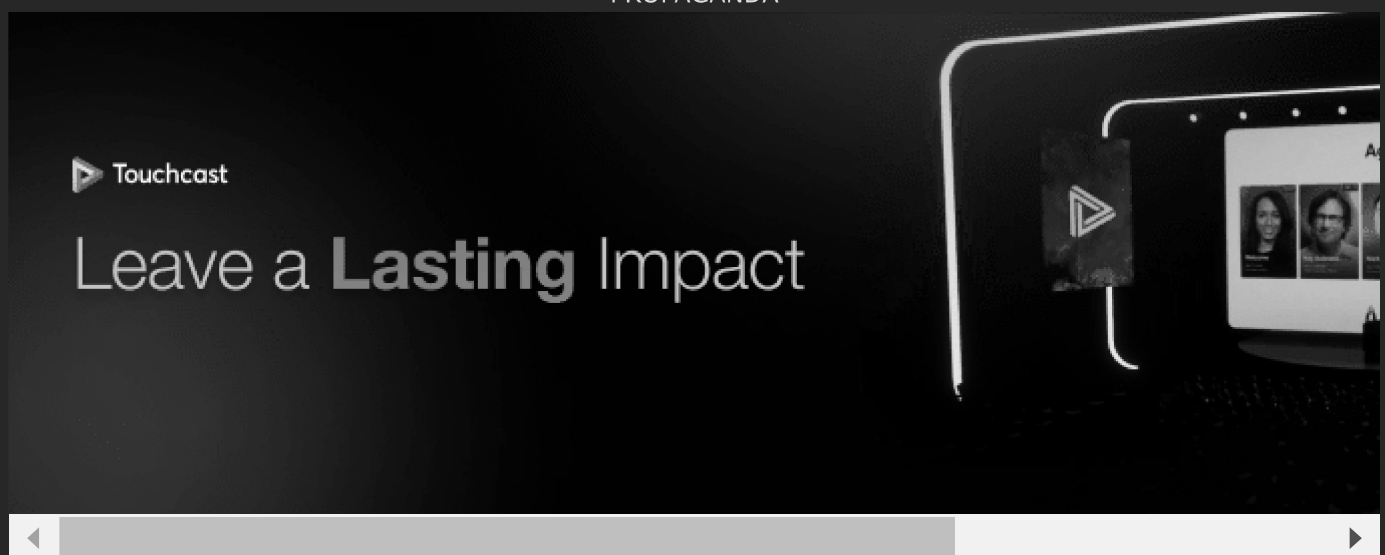
Por sorte, nosso exemplo se cadastrou todas as vezes que havia um show de comédia na cidade, e registrou algumas informações sobre o comediante, e também registrou se ele foi ou não.

Idade	Experiência	Classificação	Nacionalidade	Ir
36	10	9	Reino Unido	NÃO

42	12	4	EUA	NÃO
23	4	6	N	NÃO
52	4	4	EUA	NÃO
43	21	8	EUA	SIM
44	14	5	Reino Unido	NÃO
66	3	7	N	SIM
35	14	9	Reino Unido	SIM
52	13	7	N	SIM
35	5	9	N	SIM
24	3	5	EUA	NÃO
18	3	7	Reino Unido	SIM
45	9	9	Reino Unido	SIM

Now, based on this data set, Python can create a decision tree that can be used to decide if any new shows are worth attending to.

PROPAGANDA



How Does it Work?

First, import the modules you need, and read the dataset with pandas:

Example

Read and print the data set:

```
import pandas
from sklearn import tree
import pydotplus
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt
import matplotlib.image as pltimg

df = pandas.read_csv("shows.csv")

print(df)
```

Run example »

To make a decision tree, all data has to be numerical.

We have to convert the non numerical columns 'Nationality' and 'Go' into numerical values.

Pandas has a `map()` method that takes a dictionary with information on how to convert the values.

```
{'UK': 0, 'USA': 1, 'N': 2}
```

Means convert the values 'UK' to 0, 'USA' to 1, and 'N' to 2.

Example

Change string values into numerical values:

```
d = {'UK': 0, 'USA': 1, 'N': 2}
df['Nationality'] = df['Nationality'].map(d)
d = {'YES': 1, 'NO': 0}
df['Go'] = df['Go'].map(d)

print(df)
```

Run example »

Then we have to separate the *feature* columns from the *target* column.

The feature columns are the columns that we try to predict *from*, and the target column is the column with the values we try to predict.

Example

`X` is the feature columns, `y` is the target column:

```
features = ['Age', 'Experience', 'Rank', 'Nationality']

X = df[features]
y = df['Go']

print(X)
print(y)
```

Run example »

Now we can create the actual decision tree, fit it with our details, and save a .png file on the computer:

Example

Create a Decision Tree, save it as an image, and show the image:

```
dtree = DecisionTreeClassifier()
dtree = dtree.fit(X, y)
data = tree.export_graphviz(dtree, out_file=None, feature_names=features)
graph = pydotplus.graph_from_dot_data(data)
graph.write_png('mydecisiontree.png')

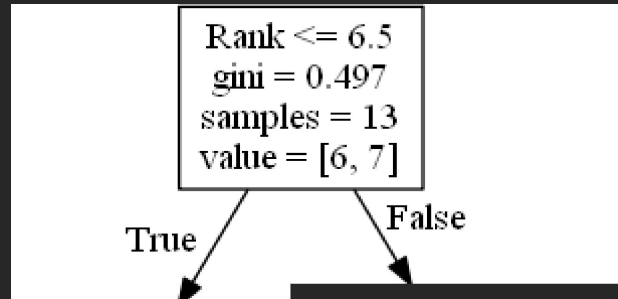
img=pltimg.imread('mydecisiontree.png')
imgplot = plt.imshow(img)
plt.show()
```

Run example »

Result Explained

The decision tree uses your earlier decisions to calculate the odds for you to wanting to go see a comedian or not.

Let us read the different aspects of the decision tree:



Rank

`Rank <= 6.5` means that every comedian with a rank of 6.5 or lower will follow the `True` arrow (to the left), and the rest will follow the `False` arrow (to the right).

`gini = 0.497` refers to the quality of the split, and is always a number between 0.0 and 0.5, where 0.0 would mean all of the samples got the same result, and 0.5 would mean that the split is done exactly in the middle.

`samples = 13` means that there are 13 comedians left at this point in the decision, which is all of them since this is the first step.

`value = [6, 7]` means that of these 13 comedians, 6 will get a "NO", and 7 will get a "GO".

Gini

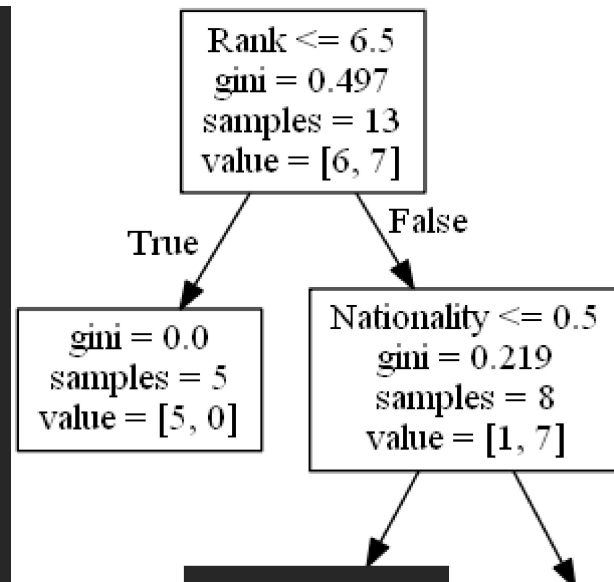
There are many ways to split the samples, we use the GINI method in this tutorial.

The Gini method uses this formula:

$$\text{Gini} = 1 - (x/n)^2 - (y/n)^2$$

Where x is the number of positive answers("GO"), n is the number of samples, and y is the number of negative answers ("NO"), which gives us this calculation:

$$1 - (7 / 13)^2 - (6 / 13)^2 = 0.497$$



The next step contains two boxes, one box for the comedians with a 'Rank' of 6.5 or lower, and one box with the rest.

True - 5 Comedians End Here:

`gini = 0.0` means all of the samples got the same result.

`samples = 5` means that there are 5 comedians left in this branch (5 comedian with a Rank of 6.5 or lower).

`value = [5, 0]` means that 5 will get a "NO" and 0 will get a "GO".

False - 8 Comedians Continue:

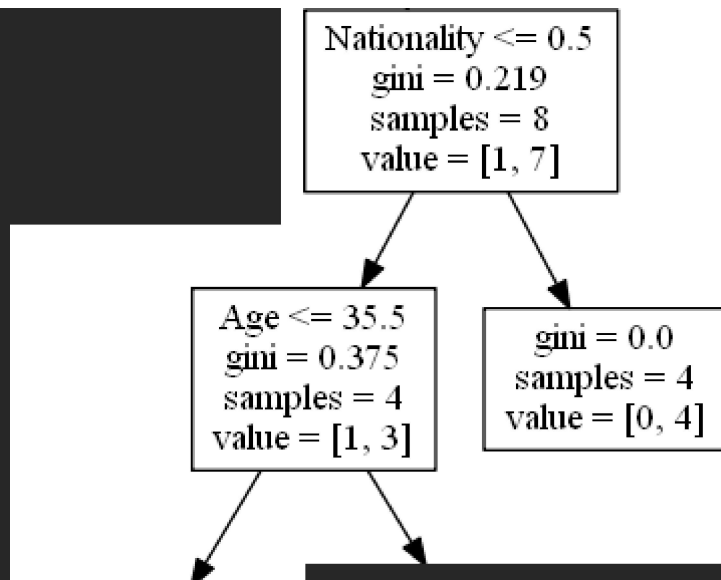
Nationality

`Nationality <= 0.5` means that the comedians with a nationality value of less than 0.5 will follow the arrow to the left (which means everyone from the UK,), and the rest will follow the arrow to the right.

`gini = 0.219` means that about 22% of the samples would go in one direction.

`samples = 8` means that there are 8 comedians left in this branch (8 comedian with a Rank higher than 6.5).

`value = [1, 7]` means that of these 8 comedians, 1 will get a "NO" and 7 will get a "GO".



True - 4 Comedians Continue:

Age

`Age <= 35.5` means that comedians at the age of 35.5 or younger will follow the arrow to the left, and the rest will follow the arrow to the right.

`gini = 0.375` means that about 37,5% of the samples would go in one direction.

`samples = 4` means that there are 4 comedians left in this branch (4 comedians from the UK).

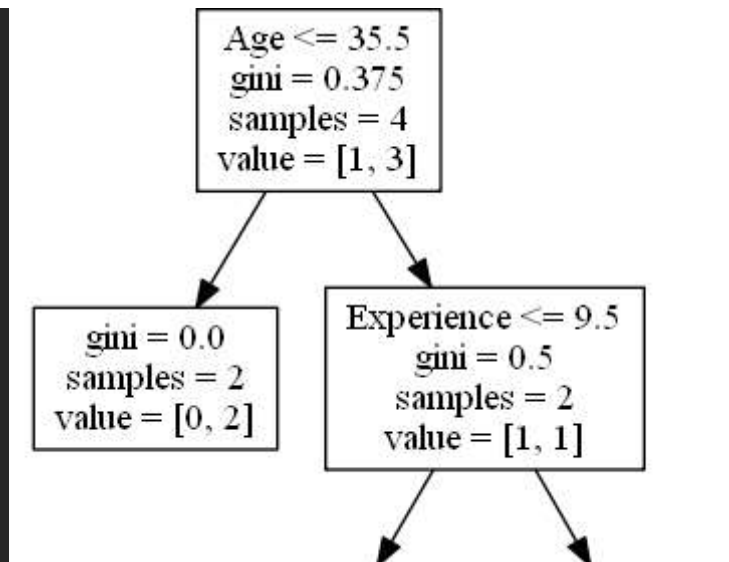
`value = [1, 3]` means that of these 4 comedians, 1 will get a "NO" and 3 will get a "GO".

False - 4 Comedians End Here:

`gini = 0.0` means all of the samples got the same result.

`samples = 4` means that there are 4 comedians left in this branch (4 comedians not from the UK).

`value = [0, 4]` means that of these 4 comedians, 0 will get a "NO" and 4 will get a "GO".



True - 2 Comedians End Here:

`gini = 0.0` means all of the samples got the same result.

`samples = 2` means that there are 2 comedians left in this branch (2 comedians at the age 35.5 or younger).

`value = [0, 2]` means that of these 2 comedians, 0 will get a "NO" and 2 will get a "GO".

False - 2 Comedians Continue:

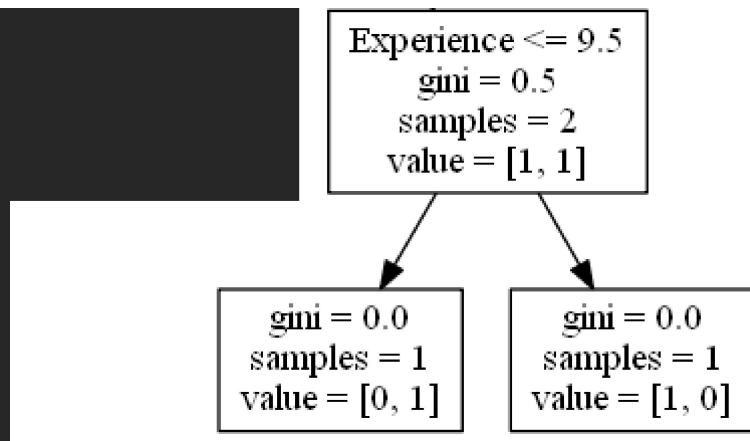
Experience

`Experience <= 9.5` means that comedians with 9.5 years of experience, or less, will follow the arrow to the left, and the rest will follow the arrow to the right.

`gini = 0.5` means that 50% of the samples would go in one direction.

`samples = 2` means that there are 2 comedians left in this branch (2 comedians older than 35.5).

`value = [1, 1]` means that of these 2 comedians, 1 will get a "NO" and 1 will get a "GO".



True - 1 Comedian Ends Here:

`gini = 0.0` means all of the samples got the same result.

`samples = 1` means that there is 1 comedian left in this branch (1 comedian with 9.5 years of experience or less).

`value = [0, 1]` means that 0 will get a "NO" and 1 will get a "GO".

False - 1 Comedian Ends Here:

`gini = 0.0` means all of the samples got the same result.

`samples = 1` significa que resta 1 comediante neste ramo (1 comediante com mais de 9,5 anos de experiência).

`value = [1, 0]` significa que 1 receberá um "NO" e 0 receberá um "GO".

Prever valores

Podemos usar a Árvore de Decisão para prever novos valores.

Exemplo: Devo ir ver um show estrelado por um comediante americano de 40 anos, com 10 anos de experiência e um ranking de comédia de 7?

Exemplo

Use o método `predict()` para prever novos valores:

```
print(dtrees.predict([[40, 10, 7, 1]]))
```

Executar exemplo »

Exemplo

Qual seria a resposta se a classificação de comédia fosse 6?

```
print(dtree.predict([[40, 10, 6, 1]]))
```

Executar exemplo »

Resultados Diferentes


Você verá que a Árvore de Decisão fornece resultados diferentes se você a executar várias vezes, mesmo que a alimente com os mesmos dados.

Isso porque a Árvore de Decisão não nos dá uma resposta 100% certa. É baseado na probabilidade de um resultado, e a resposta irá variar.

◀ Anterior


Próximo ▶

PROPAGANDA




**Don't want
to be a
victim of
ransomware
attacks?
Get Protected!**

**Get Your
Free Guide**



NOVO

Acabamos de lançar
vídeos do W3Schools



Explore now

COLOR PICKER





Get certified
by completing
a course today!



Get started

CODE GAME



Play Game

PROPAGANDA



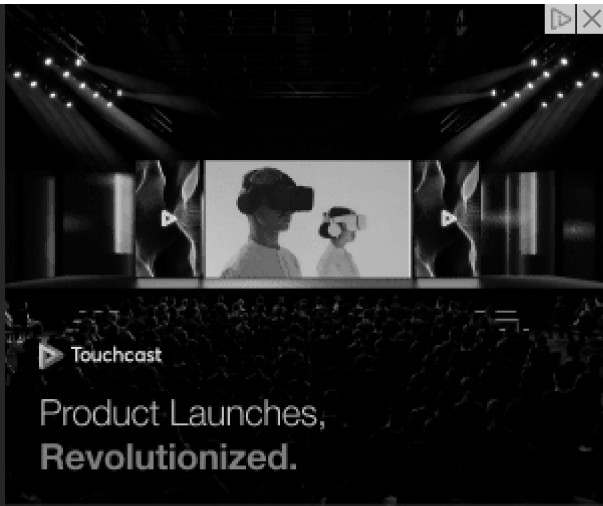
PROPAGANDA

Bocconi



REGISTER NOW!

PROPAGANDA

[Report Error](#)[Forum](#)[About](#)[Shop](#)

Top Tutorials

- [HTML Tutorial](#)
- [CSS Tutorial](#)
- [JavaScript Tutorial](#)
- [How To Tutorial](#)
- [SQL Tutorial](#)
- [Python Tutorial](#)
- [W3.CSS Tutorial](#)
- [Bootstrap Tutorial](#)
- [PHP Tutorial](#)
- [Java Tutorial](#)
- [C++ Tutorial](#)
- [jQuery Tutorial](#)

Top References

- [HTML Reference](#)
- [CSS Reference](#)
- [JavaScript Reference](#)
- [SQL Reference](#)
- [Python Reference](#)
- [W3.CSS Reference](#)
- [Bootstrap Reference](#)
- [PHP Reference](#)
- [HTML Colors](#)
- [Java Reference](#)

[Angular Reference](#)[jQuery Reference](#)

Top Examples

[HTML Examples](#)[CSS Examples](#)[JavaScript Examples](#)[How To Examples](#)[SQL Examples](#)[Python Examples](#)[W3.CSS Examples](#)[Bootstrap Examples](#)[PHP Examples](#)[Java Examples](#)[XML Examples](#)[jQuery Examples](#)

Web Courses

[HTML Course](#)[CSS Course](#)[JavaScript Course](#)[Front End Course](#)[SQL Course](#)[Python Course](#)[PHP Course](#)[jQuery Course](#)[Java Course](#)[C++ Course](#)[C# Course](#)[XML Course](#)[Get Certified »](#)

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using W3Schools, you agree to have read and accepted our [terms of use](#), [cookie and privacy policy](#).

Copyright 1999-2022 by Refsnes Data. All Rights Reserved.

W3Schools is Powered by W3.CSS.

