

How do you plot a ROC curve in Python step-by-step?

Authored by
stats writer

April 28, 2024

RECOMMENDED CITATION

stats writer (2024). *How do you plot a ROC curve in Python step-by-step?*.

PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=140329>

The process of plotting a ROC (Receiver Operating Characteristic) curve in Python involves a series of steps. First, the data must be prepared by dividing it into a training and testing set. Next, a classification model such as logistic regression or decision trees is trained using the training set. Then, the model's predictions for the testing set are collected. These predictions, along with the actual labels, are used to calculate the True Positive Rate (TPR) and False Positive Rate (FPR) for different classification thresholds. These rates are then plotted on a graph, with TPR on the y-axis and FPR on the x-axis, to create the ROC curve. Finally, the area under the curve (AUC) is calculated to evaluate the performance of the model. This step-by-step process allows for the visualization and assessment of a classification model's predictive power and can be easily implemented in Python using various libraries such as scikit-learn and matplotlib.

Plot a ROC Curve in Python (Step-by-Step)

Logistic Regression is a statistical method that we use to fit a regression model when the response variable is binary. To assess how well a logistic regression model fits a dataset, we can look at the following two metrics:

Sensitivity: The probability that the model predicts a positive outcome for an observation when indeed the outcome is positive. This is also called the "true positive rate."**Specificity:** The probability that the model predicts a negative outcome for an observation when indeed the outcome is negative. This is also called the "true negative rate."

One way to visualize these two metrics is by creating a ROC curve, which stands for "receiver operating characteristic" curve. This is a plot that displays the

sensitivity and specificity of a logistic regression model.

The following step-by-step example shows how to create and interpret a ROC curve in Python.

Step 1: Import Necessary Packages

First, we'll import the packages necessary to perform logistic regression in Python:

```
import pandas as pd  
import numpy as np  
from sklearn.model_selection import train_test_split  
from sklearn.linear_model import LogisticRegression  
from sklearn import metrics  
import matplotlib.pyplot as plt
```

Step 2: Fit the Logistic Regression Model

Next, we'll import a dataset and fit a logistic regression model to it:

```
#import dataset from CSV file on Github  
url =  
"https://raw.githubusercontent.com/Statology/Python-G
```

```
uides/main/default.csv"
data = pd.read_csv(url)

#define the predictor variables and the response
variable
X = data[
y = data

#split the dataset into training (70%) and testing (30%)
sets
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.3,random_state=0)

#instantiate the model
log_regression = LogisticRegression()

#fit the model using the training data
log_regression.fit(X_train,y_train)
```

Step 3: Plot the ROC Curve

Next, we'll calculate the true positive rate and the false positive rate and create a ROC curve using the Matplotlib data visualization package:

```
#define metrics
```

```
y_pred_proba = log_regression.predict_proba(X_test)  
fpr, tpr, _ = metrics.roc_curve(y_test, y_pred_proba)
```

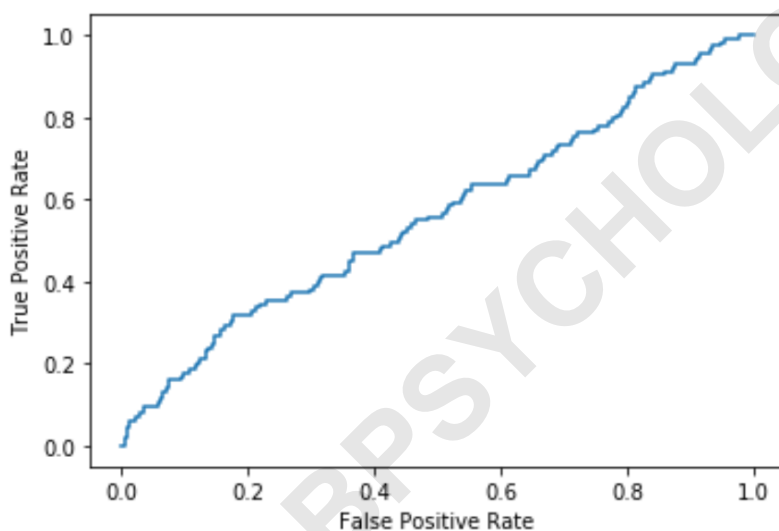
```
#create ROC curve
```

```
plt.plot(fpr,tpr)
```

```
plt.ylabel('True Positive Rate')
```

```
plt.xlabel('False Positive Rate')
```

```
plt.show()
```



The more that the curve hugs the top left corner of the plot, the better the model does at classifying the data into categories.

As we can see from the plot above, this logistic regression model does a pretty poor job of classifying the data into categories.

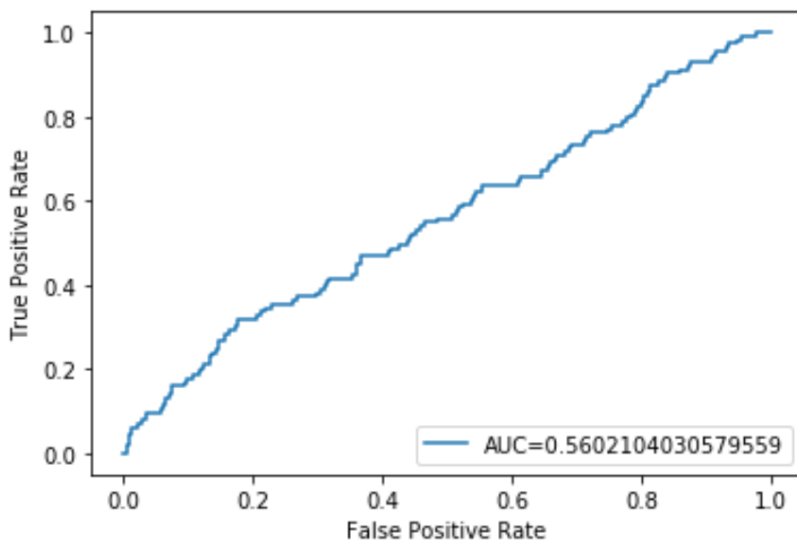
To quantify this, we can calculate the AUC - area under the curve - which tells us how much of the plot is located under the curve.

Step 4: Calculate the AUC

We can use the following code to calculate the AUC of the model and display it in the lower right corner of the ROC plot:

```
#define metrics
y_pred_proba = log_regression.predict_proba(X_test)
fpr, tpr, _ = metrics.roc_curve(y_test, y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)

#create ROC curve
plt.plot(fpr,tpr,label="AUC="+str(auc))
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.legend(loc=4)
plt.show()
```



The AUC for this logistic regression model turns out to be 0.5602. Since this is close to 0.5, this confirms that the model does a poor job of classifying data.

How to Plot Multiple ROC Curves in Python