**AI人工智慧 監督學習(分類)**

**AI人工智慧 在Python中構建分類器的步驟**

在本章中，我們將重點討論監督式學習 - **分類**。

分類技術或模型試圖從觀測值中得出一些結論。 在分類問題中，我們有分類輸出，如“黑色”或“白色”或“教學”和“非教學”。 在構建分類模型時，需要有包含資料點和相應標籤的訓練資料集。 例如，如果想檢查圖像是否屬於汽車。 要實現這個檢查，我們將建立一個訓練資料集，其中包含與“車”和“無車”相關的兩個類。 然後需要使用訓練樣本來訓練模型。 分類模型主要用於人臉識別，垃圾郵件識別等。

為了在 Python 中構建分類器，將使用 Python 3 和 Scikit-learn，這是一個用於機器學習的工具。 按照以下步驟在 Python 中構建分類器 -

**第1步** - 導入 Scikit-learn 這將是在 Python 中構建分類器的第一步。 在這一步中，將安裝一個名為 Scikit-learn 的 Python包，它是 Python中最好的機器學習模組之一。 以下命令導入包 -

1. import sklearn

**第2步** - 導入 Scikit-learn 的資料集

在這一步中，我們可以開始使用機器學習模型的資料集。 在這裡，將使用[乳腺癌威斯康辛診斷資料庫](http://scikit-learn.org/stable/datasets/index.html" \l "breast-cancer-wisconsin-diagnostic-database" \t "_blank)。 資料集包括有關乳腺癌腫瘤的各種資訊，以及惡性或良性分類標籤。 該資料集在 569 個腫瘤上具有 569 個實例或資料，並且包括關於 30 個屬性或特徵(諸如腫瘤的半徑，紋理，光滑度和面積)的資訊。 借助以下命令，導入 Scikit-learn的乳腺癌資料集 -

1. from sklearn.datasets import load\_breast\_cancer

現在，以下命令將載入資料集。

1. data = load\_breast\_cancer()

以下是字典鍵清單 -

* 分類標籤名稱(target\_names)
* 實際標籤(目標)
* 屬性/功能名稱(feature\_names)
* 屬性(資料)

現在，使用以下命令，可以為每個重要資訊集創建新變數並分配資料。 換句話說，可以用下列命令組織資料 -

1. label\_names = data['target\_names']
2. labels = data['target']
3. feature\_names = data['feature\_names']
4. features = data['data']

現在，為了使它更清晰，可以使用以下命令來列印類標籤，第一個資料實例的標籤，我們的功能名稱和功能的值 -

1. print(label\_names)

上述命令將分別列印惡性和良性的分類名稱。輸出結果如下 -

1. ['malignant' 'benign']

現在，下面的命令將顯示它們被映射到二進位值 0 和 1。這裡 0 表示惡性腫瘤，1 表示良性癌症。得到以下輸出 -

1. print(labels[0])
2. 0

下面給出的兩個命令將生成功能名稱和功能值。

1. print(feature\_names[0])
2. mean radius
3. print(features[0])
4. [ 1.79900000e+01 1.03800000e+01 1.22800000e+02 1.00100000e+03
5. 1.18400000e-01 2.77600000e-01 3.00100000e-01 1.47100000e-01
6. 2.41900000e-01 7.87100000e-02 1.09500000e+00 9.05300000e-01
7. 8.58900000e+00 1.53400000e+02 6.39900000e-03 4.90400000e-02
8. 5.37300000e-02 1.58700000e-02 3.00300000e-02 6.19300000e-03
9. 2.53800000e+01 1.73300000e+01 1.84600000e+02 2.01900000e+03
10. 1.62200000e-01 6.65600000e-01 7.11900000e-01 2.65400000e-01
11. 4.60100000e-01 1.18900000e-01]

從上面的輸出中，可以看到第一個資料實例是一個半徑為 1.7990000e + 01 的惡性腫瘤。

**第3步** - 組織資料

在這一步中，將把資料分成兩部分，即訓練集和測試集。 將資料分割成這些集合非常重要，因為必須在未看到的資料上測試模型。要將資料分成集合，sklearn 有一個叫做 train\_test\_split() 函數的函數。 在以下命令的幫助下，可以分割這些集合中的資料 -

1. from sklearn.model\_selection import train\_test\_split

上述命令將從 sklearn 中導入 train\_test\_split 函數，下面的命令將資料分解為訓練和測試資料。 在下面給出的例子中，使用 40% 的資料進行測試，其餘資料將用於訓練模型。

1. train, test, train\_labels, test\_labels = train\_test\_split(features,labels,test\_size = 0.40, random\_state = 42)

**第4步** - 建立模型 在這一步中，我們將建立模型。使用樸素貝葉斯演算法來構建模型。 以下命令可用於構建模型 -

1. from sklearn.naive\_bayes import GaussianNB

上述命令將導入 GaussianNB 模組。 現在，以下命令用來初始化模型。

1. gnb = GaussianNB()

將通過使用 gnb.fit() 將它擬合到資料來訓練模型。

1. model = gnb.fit(train, train\_labels)

**第5步** - 評估模型及其準確性 在這一步中，我們將通過對測試資料進行預測來評估模型。為了做出預測，我們將使用 predict() 函數。 以下命令做到這一點 -

1. preds = gnb.predict(test)
2. print(preds)
3. ## -- 結果如下
4. [1 0 0 1 1 0 0 0 1 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 1 1
5. 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0
6. 0 1 1 0 0 1 1 1 0 0 1 1 0 0 1 0 1 1 1 1 1 1 0 1 1 0 0 0 0
7. 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 0 0 0
8. 1 1 1 0 0 1 1 0 1 0 0 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 1 0
9. 1 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0
10. 1 1 0 1 1 1 1 1 1 0 0 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 1 1 0
11. 1 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1]

上述 0 和 1 系列是腫瘤類別的預測值 - 惡性和良性。

現在，通過比較兩個陣列即 test\_labels和 preds，可以發現模型的準確性。使用 accuracy\_score() 函數來確定準確性。 考慮下面的命令 -

1. from sklearn.metrics import accuracy\_score
2. print(accuracy\_score(test\_labels,preds))
3. 0.951754385965

結果顯示 NaïveBayes 分類器準確率為 95.17%。

通過這種方式，借助上述步驟，我們可以使用 Python 構建分類器。

**AI人工智慧 在Python中構建分類器**

2021-03-12 14:36 更新

在本節中，我們將學習如何在 Python 中構建分類器。

**樸素貝葉斯分類器**

樸素貝葉斯是一種使用貝葉斯定理建立分類器的分類技術。 假設是預測變數是獨立的。 簡而言之，它假設類中某個特徵的存在與任何其他特徵的存在無關。要構建樸素貝葉斯分類器，我們需要使用名為 *scikit learn* 的 python庫。 在 scikit 學習包中，有三種類型的樸素貝葉斯模型被稱為 Gaussian，Multinomial 和 Bernoulli。

要構建樸素貝葉斯機器學習分類器模型，需要以下“減號”

**資料集**

我們將使用名為 Breast Cancer Wisconsin Diagnostic Database資料集\_blank。 資料集包括有關乳腺癌腫瘤的各種資訊，以及惡性或良性分類標籤。 該資料集在 569 個腫瘤上具有 569 個實例或資料，並且包括關於 30 個屬性或特徵(諸如腫瘤的半徑，紋理，光滑度和面積)的資訊。可以從 sklearn 包中導入這個資料集。

**樸素貝葉斯模型**

為了構建樸素貝葉斯分類器，需要一個樸素貝葉斯模型。 如前所述，scikit 學習包中有三種類型的 NaïveBayes 模型，分別稱為 Gaussian，Multinomial 和 Bernoulli。 在下面的例子中，將使用高斯樸素貝葉斯模型。

通過使用上述內容，我們將建立一個樸素貝葉斯機器學習模型來使用腫瘤資訊來預測腫瘤是否是惡性的或良性的。

首先，我們需要安裝 sklearn 模組。 它可以通過以下命令完成 -

1. import sklearn

現在，需要導入名為 Breast Cancer Wisconsin Diagnostic Database 的資料集。

1. from sklearn.datasets import load\_breast\_cancer

現在，以下命令將載入資料集。

1. data = load\_breast\_cancer()

資料可以按如下方式組織 -

1. label\_names = data['target\_names']
2. labels = data['target']
3. feature\_names = data['feature\_names']
4. features = data['data']

現在，為了使它更清晰，可以在以下命令的幫助下列印類標籤，第一個資料實例的標籤，功能名稱和功能的值 -

1. print(label\_names)

上述命令將分別列印惡性和良性的類名。 它顯示為下面的輸出 -

1. ['malignant' 'benign']

現在，下面給出的命令將顯示它們映射到二進位值 0 和 1。這裡 0 表示惡性腫瘤，1 表示良性癌症。 它顯示為下面的輸出 -

1. print(labels[0])
2. 0

以下兩個命令將生成功能名稱和功能值。

1. print(feature\_names[0])
2. mean radius
3. print(features[0])
4. [ 1.79900000e+01 1.03800000e+01 1.22800000e+02 1.00100000e+03
5. 1.18400000e-01 2.77600000e-01 3.00100000e-01 1.47100000e-01
6. 2.41900000e-01 7.87100000e-02 1.09500000e+00 9.05300000e-01
7. 8.58900000e+00 1.53400000e+02 6.39900000e-03 4.90400000e-02
8. 5.37300000e-02 1.58700000e-02 3.00300000e-02 6.19300000e-03
9. 2.53800000e+01 1.73300000e+01 1.84600000e+02 2.01900000e+03
10. 1.62200000e-01 6.65600000e-01 7.11900000e-01 2.65400000e-01
11. 4.60100000e-01 1.18900000e-01]

從以上輸出可以看出，第一個資料實例是一個主要半徑為 1.7990000e + 01 的惡性腫瘤。

要在未看到的資料上測試模型，我們需要將資料分解為訓練和測試資料。 它可以在下面的代碼的幫助下完成 -

1. from sklearn.model\_selection import train\_test\_split

上述命令將從 sklearn 中導入 train\_test\_split 函數，下面的命令將資料分解為訓練和測試資料。 在下面的例子中，使用 40%的資料進行測試，並將提示資料用於訓練模型。

1. train, test, train\_labels, test\_labels =
2. train\_test\_split(features,labels,test\_size = 0.40, random\_state = 42)

現在，使用以下命令構建模型 -

1. from sklearn.naive\_bayes import GaussianNB

上述命令將從 sklearn 中導入train\_test\_split 函數，下面的命令將資料分解為訓練和測試資料。 在下面的例子中，我們使用 40% 的資料進行測試，並將提示資料用於訓練模型。

1. train, test, train\_labels, test\_labels =
2. train\_test\_split(features,labels,test\_size = 0.40, random\_state = 42)

現在，使用以下命令構建模型 -

1. from sklearn.naive\_bayes import GaussianNB

上述命令將導入 GaussianNB 模組。 現在，使用下面給出的命令，需要初始化模型。

1. gnb = GaussianNB()

將通過使用 gnb.fit()將它擬合到資料來訓練模型。

1. model = gnb.fit(train, train\_labels)

現在，通過對測試資料進行預測來評估模型，並且可以按如下方式完成 -

1. preds = gnb.predict(test)
2. print(preds)
3. [1 0 0 1 1 0 0 0 1 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 1 1 1 1 1
4. 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0
5. 0 1 1 0 0 1 1 1 0 0 1 1 0 0 1 0 1 1 1 1 1 1 0 1 1 0 0 0 0
6. 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0 1 1 0 0 0
7. 1 1 1 0 0 1 1 0 1 0 0 1 1 0 0 0 1 1 1 0 1 1 0 0 1 0 1 1 0
8. 1 0 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0
9. 1 1 0 1 1 1 1 1 1 0 0 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 1 1 0
10. 1 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1]

上述0和1系列是腫瘤類別的預測值，即惡性和良性。

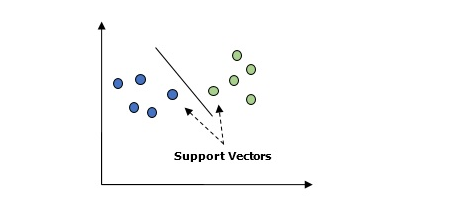
現在，通過比較兩個陣列即 test\_labels 和 preds，可以看到模型的準確性。 我們將使用 accuracy\_score() 函數來確定準確性。 考慮下面的命令 -

1. from sklearn.metrics import accuracy\_score
2. print(accuracy\_score(test\_labels,preds))
3. 0.951754385965

結果顯示 NaïveBayes 分類器準確率為 95.17%。

這是基於 NaïveBayse 高斯模型的機器學習分類器。

**支持向量機(SVM)** 基本上，支持向量機(SVM)是一種有監督的機器學習演算法，可用於回歸和分類。 SVM 的主要概念是將每個資料項目繪製為n維空間中的一個點，每個特徵的值是特定座標的值。以下是瞭解 SVM 概念的簡單圖形表示 -



在上圖中，有兩個特徵。 因此，首先需要在二維空間中繪製這兩個變數，其中每個點都有兩個座標，稱為支援向量。 該行將資料分成兩個不同的分類組。 這條線將是分類器。

在這裡，將使用 scikit-learn 和 iris 資料集來構建 SVM 分類器。 Scikitlearn 庫具有 sklearn.svm 模組並提供 sklearn.svm.svc 進行分類。 下面顯示了基於 4 個特徵來預測虹膜植物種類的 SVM 分類器。

**資料集**

我們將使用包含3個類別(每個類別為 50 個實例)的虹膜資料集，其中每個類別指的是一類虹膜工廠。 每個實例具有四個特徵，即萼片長度，萼片寬度，花瓣長度和花瓣寬度。 下面顯示了基於4個特徵來預測虹膜植物分類的 SVM 分類器。

**內核** 這是 SVM 使用的技術。 基本上這些功能採用低維輸入空間並將其轉換到更高維空間。 它將不可分離的問題轉換成可分離的問題。 核函數可以是線性，多項式，rbf 和 sigmoid 中的任何一種。 在這個例子中，將使用線性內核。

現在導入下列套裝軟體 -

1. import pandas as pd
2. import numpy as np
3. from sklearn import svm, datasets
4. import matplotlib.pyplot as plt

現在，載入輸入資料 -

1. iris = datasets.load\_iris()

我們使用前兩個功能 -

1. X = iris.data[:, :2]
2. y = iris.target

我們將用原始資料繪製支援向量機邊界，創建一個網格來繪製。

1. x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1
2. y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1
3. h = (x\_max / x\_min)/100
4. xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h),
5. np.arange(y\_min, y\_max, h))
6. X\_plot = np.c\_[xx.ravel(), yy.ravel()]

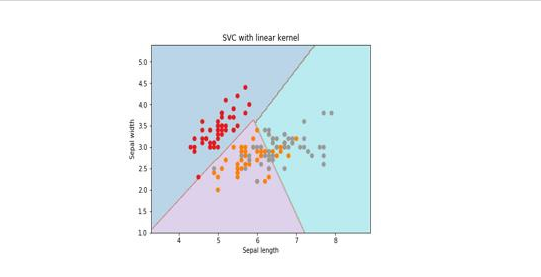
需要給出正則化參數的值。

1. C = 1.0

需要創建 SVM 分類器物件。參考以下代碼 -

1. Svc\_classifier = svm\_classifier.SVC(kernel='linear',
2. C=C, decision\_function\_shape = 'ovr').fit(X, y)
3. Z = svc\_classifier.predict(X\_plot)
4. Z = Z.reshape(xx.shape)
5. plt.figure(figsize = (15, 5))
6. plt.subplot(121)
7. plt.contourf(xx, yy, Z, cmap = plt.cm.tab10, alpha = 0.3)
8. plt.scatter(X[:, 0], X[:, 1], c = y, cmap = plt.cm.Set1)
9. plt.xlabel('Sepal length')
10. plt.ylabel('Sepal width')
11. plt.xlim(xx.min(), xx.max())
12. plt.title('SVC with linear kernel')

執行後得到以下結果 -



**AI人工智慧 邏輯回歸**

2020-09-23 15:28 更新

基本上，邏輯回歸模型是監督分類演算法族的成員之一。 Logistic 回歸通過使用邏輯函數估計概率來測量因變數和引數之間的關係。

在這裡，如果我們討論依賴變數和獨立變數，那麼因變數就是要預測的目標類變數，另一方面，引數是用來預測目標類的特徵。

在邏輯回歸中，估計概率意味著預測事件的可能性發生。例如，店主想要預測進入商店的顧客將購買遊戲站(例如)或不購買。顧客將會觀察到許多顧客的特徵 - 性別，年齡等，以便預測可能性的發生，即購買遊戲站或不購物。邏輯函數是用來構建具有各種參數的函數的 S 形曲線。

**前提條件**

在使用邏輯回歸構建分類器之前，我們需要在系統上安裝 Tkinter 套裝軟體。 它可以從 [此處](https://docs.python.org/2/library/tkinter.html) 進行安裝。

現在，在下面給出的代碼的説明下，可以使用邏輯回歸來創建分類器 -

首先，導入一些套裝軟體 -

1. import numpy as np
2. from sklearn import linear\_model
3. import matplotlib.pyplot as plt

現在，需要定義可以完成的樣本資料，如下所示 -

1. X = np.array([[2, 4.8], [2.9, 4.7], [2.5, 5], [3.2, 5.5], [6, 5], [7.6, 4],
2. [3.2, 0.9], [2.9, 1.9],[2.4, 3.5], [0.5, 3.4], [1, 4], [0.9, 5.9]])
3. y = np.array([0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3])

接下來，我們需要創建邏輯回歸分類器，可以按如下方式完成 -

1. Classifier\_LR = linear\_model.LogisticRegression(solver = 'liblinear', C = 75)

最後但重要的是，我們需要訓練這個分類器 -

1. Classifier\_LR.fit(X, y)

現在，我們如何視覺化輸出？ 可以通過創建一個名為Logistic\_visualize()的函數來完成 -

1. Def Logistic\_visualize(Classifier\_LR, X, y):
2. min\_x, max\_x = X[:, 0].min() - 1.0, X[:, 0].max() + 1.0
3. min\_y, max\_y = X[:, 1].min() - 1.0, X[:, 1].max() + 1.0

在上面的行中，我們定義了在網格中使用的最小值和最大值 X和 Y。另外，還將定義繪製網格的步長。

1. mesh\_step\_size = 0.02

下面定義 X 和 Y 值的網格，如下所示 -

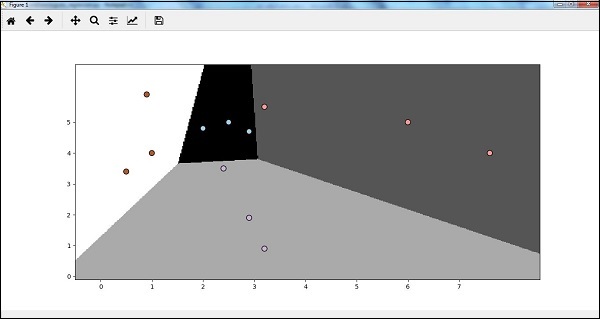
1. x\_vals, y\_vals = np.meshgrid(np.arange(min\_x, max\_x, mesh\_step\_size),
2. np.arange(min\_y, max\_y, mesh\_step\_size))

使用以下代碼，可以在網格網格上運行分類器 -

1. output = classifier.predict(np.c\_[x\_vals.ravel(), y\_vals.ravel()])
2. output = output.reshape(x\_vals.shape)
3. plt.figure()
4. plt.pcolormesh(x\_vals, y\_vals, output, cmap = plt.cm.gray)
5. plt.scatter(X[:, 0], X[:, 1], c = y, s = 75, edgecolors = 'black',
6. linewidth=1, cmap = plt.cm.Paired)

以下代碼行將指定圖的邊界 -

1. plt.xlim(x\_vals.min(), x\_vals.max())
2. plt.ylim(y\_vals.min(), y\_vals.max())
3. plt.xticks((np.arange(int(X[:, 0].min() - 1), int(X[:, 0].max() + 1), 1.0)))
4. plt.yticks((np.arange(int(X[:, 1].min() - 1), int(X[:, 1].max() + 1), 1.0)))
5. plt.show()

現在，在運行代碼之後，我們將得到以下輸出，邏輯回歸分類器 - 

**AI人工智慧 決策樹分類器**

2020-09-23 15:36 更新

決策樹基本上是一個二叉樹流程圖，其中每個節點根據某個特徵變數分割一組觀察值。

在這裡，我們正在構建一個用於預測男性或女性的決策樹分類器。這裡將採取一個非常小的資料集，有 19 個樣本。 這些樣本將包含兩個特徵 - “身高”和“頭髮長度”。

**前提條件**

為了構建以下分類器，我們需要安裝 pydotplus和 graphviz。 基本上，graphviz 是使用點檔繪製圖形的工具，pydotplus 是 Graphviz 的 Dot 語言模組。 它可以與包管理器或使用pip來安裝。

現在，可以在以下 Python 代碼的幫助下構建決策樹分類器 -

首先，導入一些重要的庫如下 -

1. import pydotplus
2. from sklearn import tree
3. from sklearn.datasets import load\_iris
4. from sklearn.metrics import classification\_report
5. from sklearn import cross\_validation
6. import collections

現在，提供如下資料集 -

1. X = [[165,19],[175,32],[136,35],[174,65],[141,28],[176,15],[131,32],
2. [166,6],[128,32],[179,10],[136,34],[186,2],[126,25],[176,28],[112,38],
3. [169,9],[171,36],[116,25],[196,25]]
4. Y = ['Man','Woman','Woman','Man','Woman','Man','Woman','Man','Woman',
5. 'Man','Woman','Man','Woman','Woman','Woman','Man','Woman','Woman','Man']
6. data\_feature\_names = ['height','length of hair']
7. X\_train, X\_test, Y\_train, Y\_test = cross\_validation.train\_test\_split
8. (X, Y, test\_size=0.40, random\_state=5)

在提供資料集之後，需要擬合可以如下完成的模型 -

1. clf = tree.DecisionTreeClassifier()
2. clf = clf.fit(X,Y)

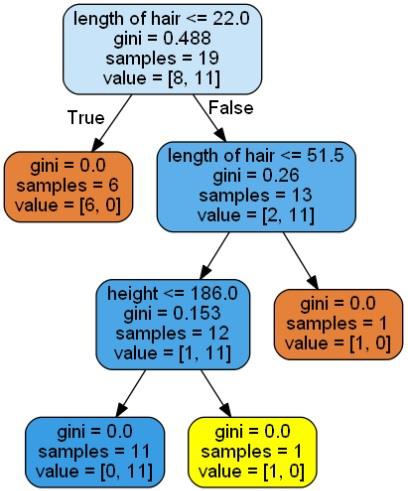
預測可以使用以下 Python 代碼來完成 -

1. prediction = clf.predict([[133,37]])
2. print(prediction)

使用以下 Python 代碼來實現視覺化決策樹 -

1. dot\_data = tree.export\_graphviz(clf,feature\_names = data\_feature\_names,
2. out\_file = None,filled = True,rounded = True)
3. graph = pydotplus.graph\_from\_dot\_data(dot\_data)
4. colors = ('orange', 'yellow')
5. edges = collections.defaultdict(list)
6. for edge in graph.get\_edge\_list():
7. edges[edge.get\_source()].append(int(edge.get\_destination()))
8. for edge in edges: edges[edge].sort()
9. for i in range(2):dest = graph.get\_node(str(edges[edge][i]))[0]
10. dest.set\_fillcolor(colors[i])
11. graph.write\_png('Decisiontree16.png')

它會將上述代碼的預測作為[‘Woman’]並創建以下決策樹 -



可以改變預測中的特徵值來測試它。

**AI人工智慧 隨機森林分類器**

2020-09-23 15:37 更新

集成方法是將機器學習模型組合成更強大的機器學習模型的方法。 隨機森林是決策樹的集合，就是其中之一。 它比單一決策樹好，因為在保留預測能力的同時，通過平均結果可以減少過度擬合。 在這裡，我們將在 scikit 學習癌症資料集上實施隨機森林模型。

導入必要的套裝軟體 -

1. from sklearn.ensemble import RandomForestClassifier
2. from sklearn.model\_selection import train\_test\_split
3. from sklearn.datasets import load\_breast\_cancer
4. cancer = load\_breast\_cancer()
5. import matplotlib.pyplot as plt
6. import numpy as np

現在，需要按照以下方式提供資料集

1. cancer = load\_breast\_cancer()
2. X\_train, X\_test, y\_train,
3. y\_test = train\_test\_split(cancer.data, cancer.target, random\_state = 0)

在提供資料集之後，需要擬合可以如下完成的模型 -

1. forest = RandomForestClassifier(n\_estimators = 50, random\_state = 0)
2. forest.fit(X\_train,y\_train)

現在，獲得訓練以及測試子集的準確性:如果增加估計器的數量，那麼測試子集的準確性也會增加。

1. print('Accuracy on the training subset:(:.3f)',format(forest.score(X\_train,y\_train)))
2. print('Accuracy on the training subset:(:.3f)',format(forest.score(X\_test,y\_test)))

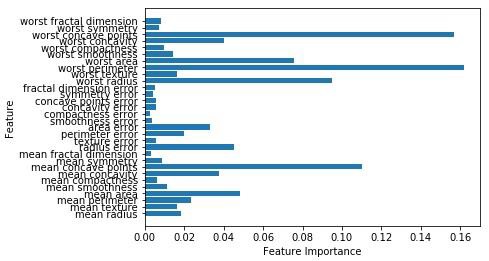
上面代碼，輸出結果如下所示 -

1. Accuracy on the training subset:(:.3f) 1.0
2. Accuracy on the training subset:(:.3f) 0.965034965034965

現在，與決策樹一樣，隨機森林具有 feature\_importance 模組，它將提供比決策樹更好的特徵權重視圖。 它可以如下繪製和視覺化 -

1. n\_features = cancer.data.shape[1]
2. plt.barh(range(n\_features),forest.feature\_importances\_, align='center')
3. plt.yticks(np.arange(n\_features),cancer.feature\_names)
4. plt.xlabel('Feature Importance')
5. plt.ylabel('Feature')
6. plt.show()

執行上面代碼，得到以下輸出結果 -



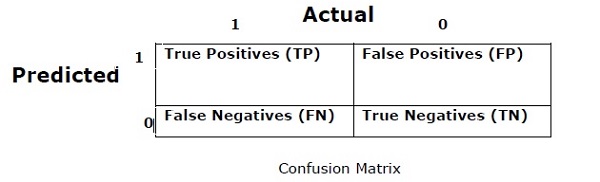
**AI人工智慧 分類器的性能**

2020-09-23 15:38 更新

在實現機器學習演算法之後，我們需要找出模型的有效性。 衡量有效性的標準可以基於資料集和度量標準。 為了評估不同的機器學習演算法，我們可以使用不同的性能指標。 例如，假設使用分類器來區分不同物件的圖像，可以使用分類性能指標，如平均準確率，AUC 等。從某種意義上說，我們選擇評估機器學習模型的指標是非常重要的，因為指標的選擇會影響機器學習演算法的性能如何被測量和比較。 以下是一些指標 -

**混亂矩陣**

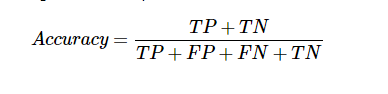
基本上它用於輸出可以是兩種或更多種類的分類問題。 這是衡量分類器性能的最簡單方法。 混淆矩陣基本上是一個包含兩個維度即“實際”和“預測”的表格。 這兩個維度都有“真正的正面(TP)”，“真正的負面(TN)”，“錯誤的正面(FP)”，“錯誤的否定(FN)”。

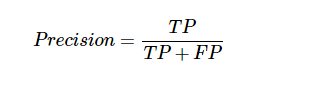


在上面的混淆矩陣中，1表示正類，0表示負類。 以下是與混淆矩陣相關的術語 -

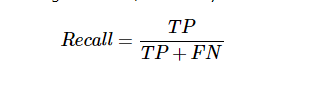
* 真正 - 當實際的資料點類別為 1 並且預測也為 1 時，TP 就是這種情況。
* 真負 - 當資料點的實際類別為 0 並且預測也為 0 時，TN 就是這種情況。
* 假正 - 當實際的資料點類別為 0 並且預測也為 1 時，FP 就是這種情況。
* 假負 - FN 是資料點的實際類別為 1 且預測也為 0的情況。

**準確性**

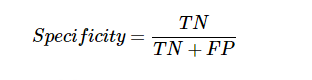
混淆矩陣本身並不是一個性能指標，但幾乎所有的性能矩陣均基於混淆矩陣。 其中之一是準確性。 在分類問題中，它可能被定義為由模型對各種預測所做的正確預測的數量。 計算準確度的公式如下 - 

**精確** 它主要用於檔檢索。 它可能被定義為返回的檔有多少是正確的。 以下是計算精度的公式 - 

**召回或靈敏度** 它可能被定義為模型返回的正數有多少。 以下是計算模型召回/靈敏度的公式 -



**特異性** 它可以定義為模型返回的負數有多少。 這與召回完全相反。 以下是計算模型特異性的公式 -



**AI人工智慧 分類失衡問題**

2020-09-23 15:40 更新

分類不平衡是屬於一個類別的觀察數量顯著低於屬於其他類別的觀測數量的場景。 例如，在我們需要識別罕見疾病，銀行欺詐性交易等情況下，這個問題非常突出。

**不平衡分類的例子** 讓我們考慮一個欺詐檢測資料集的例子來理解不平衡分類的概念 -

1. Total observations = 5000
2. Fraudulent Observations = 50
3. Non-Fraudulent Observations = 4950
4. Event Rate = 1%

**解決** 平衡類的行為，解決不平衡的類問題。 平衡類的主要目標是增加少數類的頻率或減少多數類的頻率。 以下是解決失衡類問題的方法 -

**重採樣** 重新採樣是用於重建樣本資料集的一系列方法 - 包括訓練集和測試集。 重新抽樣是為了提高模型的準確性。 以下是一些重新抽樣技術 -

* **隨機抽樣** - 這項技術旨在通過隨機排除大多數類別的例子來平衡課堂分佈。 這樣做直到大多數和少數群體的實例得到平衡。

1. Total observations = 5000
2. Fraudulent Observations = 50
3. Non-Fraudulent Observations = 4950
4. Event Rate = 1%

在這種情況下，我們將 10% 的樣本從非欺詐實例中取而代之，然後將它們與欺詐實例相結合 - 隨機抽樣後的非欺詐性觀察:4950 的 10% = 495 將他們與欺詐觀察結合後的總觀測值: 50 + 495 = 545

因此，現在，低採樣後新資料集的事件率為: 9%

這種技術的主要優點是可以減少執行時間並改善存儲。 但另一方面，它可以丟棄有用的資訊，同時減少訓練資料樣本的數量。

* **隨機抽樣** - 這種技術旨在通過複製少數類中的實例數量來平衡類分佈。

1. Total observations = 5000
2. Fraudulent Observations = 50
3. Non-Fraudulent Observations = 4950
4. Event Rate = 1%

如果複製 50 次欺詐性觀察 30 次，那麼在複製少數類別觀察值後欺詐觀察值將為 1500。然後，在過採樣後新資料中的總觀察值將為:4950 + 1500 = 6450。因此，新資料集的事件率是:1500/6450 = 23%。

這種方法的主要優點是不會丟失有用的資訊。 但另一方面，由於它複製了少數族群的事件，因此它有更多的過度機會。

**AI人工智慧 合奏技巧**

2020-09-24 13:46 更新

這種方法基本上用於修改現有的分類演算法，使其適用於不平衡的資料集。 在這種方法中，我們從原始資料中構建幾個兩階段分類器，然後匯總它們的預測。 隨機森林分類器是基於集合的分類器的一個例子。

