Kennzahlen_Klassifizierung

April 19, 2021

1 Wichtige Kennzahlen für die Klassifizierung

Hier werden die wichtigsten Kennzahlen für ein Klassifizierungsmodell vorgestellt.

Als Datensatz verwenden wir eine CSV-Datei, die direkt die Ergebnisse des Modells enthält (Feature *PROGNOSE*). In der ersten Spalte sind die realen Werte enthalten (Feature *REALITAET*).

0 entspricht hier NEGATIV 1 entspricht hier POSITIV

[31]:
```
import pandas as pd
url = "https://raw.githubusercontent.com/troescherw/datasets/master/
 ↪daten_kennzahlen_klassifizierung.csv"
df = pd.read_csv(url, delimiter=";")
df
```

[31]:

	REALITAET	PROGNOSE
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0
..	…	…
172	0	0
173	0	0
174	0	0
175	0	0
176	0	0

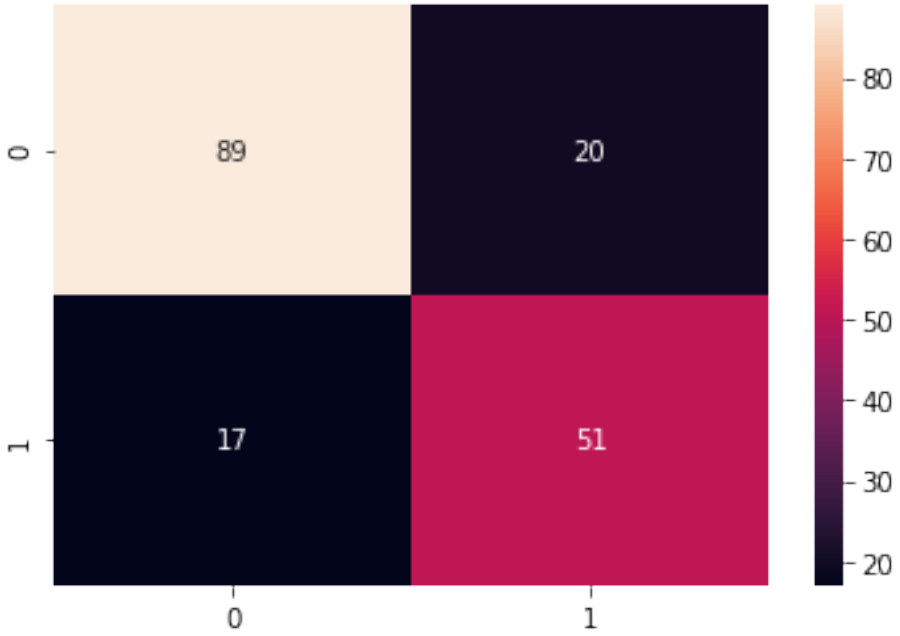
[177 rows x 2 columns]

Zuerst erstellen wir einen Report für diesen Datensatz. Mit *Seaborn* erstellen wir zudem eine Confusion Matrix mit Hilfe der Funktion *heatmap*. Man beachte hier, dass die Zeilen / Spalten gegenüber üblicher Darstellung in der Literatur vertauscht sind. Spaltenweise sind hier die Prognosen angegeben, zeilenweise die Realität.

[33]:
```
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

```
labels=["NEGATIV", "POSITIV"]
#sklearn.metrics.confusion_matrix(y_true, y_pred, *, labels=None,␣
 ↪sample_weight=None, normalize=None)[source]
cm = confusion_matrix(df.REALITAET, df.PROGNOSE)


_=sns.heatmap(cm, annot=True)
```



[28]:
```
# Report
from sklearn.metrics import classification_report

target_names = ["NEGATIV", "POSITIV"]
print(classification_report(df.REALITAET, df.PROGNOSE,␣
 ↪target_names=target_names))
```

```
              precision    recall  f1-score   support

     NEGATIV       0.84      0.82      0.83       109
     POSITIV       0.72      0.75      0.73        68

    accuracy                           0.79       177
   macro avg       0.78      0.78      0.78       177
weighted avg       0.79      0.79      0.79       177
```

Man kann natürlich jede dieser Kennzahlen (und noch viele weitere) mit Hilfe von Funktionen

berechnen lassen:

[54]:
```
from sklearn.metrics import *
TN, FP, FN, TP =  cm[0][0], cm[0][1], cm[1][0],cm[1][1]

accuracy = accuracy_score(df.REALITAET, df.PROGNOSE) # Accuracy
sensitivity = TP / (TP+FN) # Sensitivität
specificity = TN / (TN+FP) # Spezifität
precision = precision_score(df.REALITAET, df.PROGNOSE) # Precision
balanced_acc = balanced_accuracy_score(df.REALITAET, df.PROGNOSE) # balanced␣
 ↪accuracy
f1 = f1_score(df.REALITAET, df.PROGNOSE) # F1-Score
mcc = matthews_corrcoef(df.REALITAET, df.PROGNOSE) # Mattews␣
 ↪Korrelationskoeffizient

print("Accuracy = ", accuracy)
print("Sensitivitaet / Recall = ", sensitivity)
print("Spezifität = " , specificity)
print("Precision = ", precision)
print("Balanced Accuracy = " , balanced_acc)
print("F1-Score = " , f1)
print("Matthews Korrelationskoeffizient = " , mcc)
```

```
Accuracy =  0.7909604519774012
Sensitivitaet / Recall =  0.75
Spezifität =  0.8165137614678899
Precision =  0.7183098591549296
Balanced Accuracy =  0.783256880733945
F1-Score =  0.7338129496402879
Matthews Korrelationskoeffizient =  0.5622067587610047
```