UNIVERSITÄT
ZU KÖLN

SPRACHVERARBEITUNG: ÜBUNG

SoSe 2024

Janis Pagel

Institut für Digital Humanities

2024-05-28

01

ORGANIZATIONAL THINGS

Organization

- Next week on June 4th: Online (Zoom) presentation by Elke Smith from the Psychology Department of the University of Cologne: “Predicting signs of problem-gambling from online texts using large language models”
  - Zoom link will be shared on the course website `https://lehre.idh.uni-koeln.de/lehrveranstaltungen/sommersemester-2024/sprachverarbeitung/` and via Ilias mail

02

SOLUTION TO EXERCISE 05

Solution to Exercise 05

- https://lehre.idh.uni-koeln.de/site/assets/files/5151/solution05.pdf

03

DECISION TREES IN PYTHON

sklearn

- Python has a powerful library for all machine learning purposes, called `sklearn`
- Also contains a decision tree implementation: sklearn.tree.DecisionTreeClassifier

Train Dataset

Taken from the lecture slides: token features for spam and not-spam

```
import pandas as pd

data = {'casino': [1, 0, 1, 1, 0],
        'enlargement': [1, 1, 0, 1, 1],
        'meeting': [0, 0, 1, 1, 1],
        'profit': [0, 1, 0, 0, 0],
        'super': [1, 0, 1, 0, 0],
        'text': [1, 0, 0, 0, 1],
        'xxx': [1, 0, 0, 0, 1],
       'class': [0, 1, 0, 0, 1]}
df = pd.DataFrame(data)
print(df)
>    casino  enlargement  meeting  profit  super  text  xxx  class
> 0       1            1        0       0      1     1    1      0
> 1       0            1        0       1      0     0    0      1
> 2       1            0        1       0      1     0    0      0
> 3       1            1        1       0      0     0    0      0
> 4       0            1        1       0      0     1    1      1
```

Train Dataset

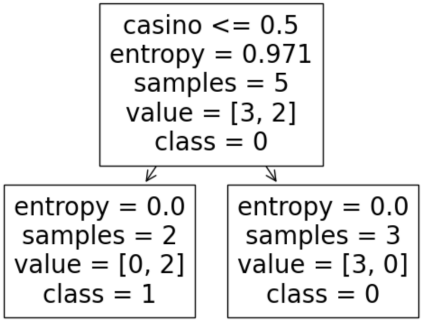
Separate the class variable (y) from the features (X) to let the classifier know what to predict

```
X = df.drop("class", axis=1)
print(X)
>    casino  enlargement  meeting  profit  super  text  xxx
> 0       1            1        0       0      1     1    1
> 1       0            1        0       1      0     0    0
> 2       1            0        1       0      1     0    0
> 3       1            1        1       0      0     0    0
> 4       0            1        1       0      0     1    1
y = df["class"]
print(y)
> 0    0
> 1    1
> 2    0
> 3    0
> 4    1
> Name: class, dtype: int64
```

Train the classifier

```
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier(criterion="entropy") # Initialize an empty classifier. criterion="entropy" is
                                                    needed to let the classifier use entropy-based
                                                    information gain (not the default)
clf = clf.fit(X,y) # The fit-function carries out the actual training and runs the algorithm using the data
                                                    provided

from sklearn.tree import plot_tree # Use the plot_tree function to visualize the tree
plot_tree(clf, feature_names = X.columns, class_names = ["0", "1"])
```



Predict on test data and evaluate

```
test_df = pd.DataFrame({'casino': [1, 0, 0],
                        'enlargement': [0, 1, 0],
                        'meeting': [0, 0, 1],
                        'profit': [0, 1, 1],
                        'super': [1, 1, 1],
                        'text': [0, 1, 1],
                        'xxx': [1, 1, 0],
                        'class': [0, 0, 1]})
print(test_df)
>    casino  enlargement  meeting  profit  super  text  xxx  class
> 0       1            0        0       0      1     0    1      0
> 1       0            1        0       1      1     1    1      0
> 2       0            0        1       1      1     1    0      1

X_test = test_df.drop("class", axis=1)
y_test = test_df["class"]
y_pred = clf.predict(X_test)
print(y_pred)
> array([0, 1, 1])

from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
>               precision    recall  f1-score   support
>
>            0       1.00      0.50      0.67         2
>            1       0.50      1.00      0.67         1
>
>     accuracy                           0.67         3
>    macro avg       0.75      0.75      0.67         3
> weighted avg       0.83      0.67      0.67         3 # "weighted avg" was called "micro avg" in the lecture slides
```

04

EXERCISE 06

Exercise 06

- https://lehre.idh.uni-koeln.de/site/assets/files/5151/exercise06.pdf


UNIVERSITY
OF COLOGNE
Janis Pagel
Institut für Digital Humanities
eMail janis.pagel@uni-koeln.de
Website https://janispagel.de
Foto: Gregor Hübl