



UNIVERSITÄT
ZU KÖLN

NAIVE BAYES

Sprachverarbeitung: Übung

Janis Pagel

Guest Lecture

- When?** Next week, May 3, 16:00
Where? In this room: Philosophikum, Seminarraum S91
Who? Johanna Binnewitt, Bundesinstitut für Berufsbildung (BIBB)
About? "Stellenanzeigen-Analyse in der sozialwissenschaftlichen Forschung"

01

SOLUTION TO EXERCISE 06

Solution to Exercise 06

- <https://lehre.idh.uni-koeln.de/site/assets/files/5615/solution06.pdf>

02

SMOOTHING



Add-One Smoothing

- In addition to the information on the [lecture slides](#), 25–26
- Add-one smoothing is special case of Laplace smoothing
- Idea: Shift around mass from the probability distribution so that 0-probabilities never occur, by adding an invented occurrence to every feature
- Example (with made-up numbers):

$$p(\text{feature1}|\text{class1}) = \frac{8}{10} \quad (1)$$

$$p(\text{feature2}|\text{class1}) = \frac{0}{10} \quad (2)$$

$$p(\text{class1}|\text{feature1}, \text{feature2}) \propto \frac{8}{10} \times \frac{0}{10} \times p(\text{class1}) = 0 \quad (3)$$

- If feature2 never occurred in the training data but occurs in an unseen sample, the probability for this class will always be 0, which is not what we want
- Solution:

$$p(\text{feature1}|\text{class1}) = \frac{8+1}{10+1} \quad (4)$$

$$p(\text{feature2}|\text{class1}) = \frac{0+1}{10+1} \quad (5)$$

$$p(\text{class1}|\text{feature1}, \text{feature2}) \propto \frac{9}{11} \times \frac{1}{11} \times p(\text{class1}) > 0 \quad (6)$$

03

NAIVE BAYES IN PYTHON

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

Train Dataset

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.naive_bayes import BernoulliNB  
clf = BernoulliNB() # Initialize an empty classifier  
clf = clf.fit(X,y) # The fit-function carries out the actual training and runs the algorithm using the data provided
```

Predict on test data and evaluate I

```
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           0.00      1.00      0.50      0.67         2
~>    1           0.50      0.50      0.50      0.67         1
```

Predict on test data and evaluate II

```
~>
~>   accuracy          0.67      3
~> macro avg    0.75    0.75    0.67      3
~> weighted avg 0.83    0.67    0.67      3
```

04

EXERCISE 07

Exercise 07

- <https://lehre.idh.uni-koeln.de/site/assets/files/5615/exercise07.pdf>



UNIVERSITY
OF COLOGNE

Janis Pagel
Institut für Digital Humanities

eMail janis.pagel@uni-koeln.de
Homepage <https://janispagel.de>
Phone +49 221 470 5749