Excursus Naive Bayes

Excursus - Naive Bayes

- “supervised learning algorithm” (https://scikit-learn.org/stable/modules/naive_bayes.html)

Excursus - Naive Bayes

- Based on Bayes' theorem:

  “the probability of an event, based on prior knowledge of conditions that might be related to the event” (https://en.wikipedia.org/wiki/Bayes%27_theorem)

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Tennis

Features
# Outlook
Sunny=0, Overcast=1, Rain=2
# Temperature:
Hot=0, Mild=1, Cool=2
# Humidity
High=0, Normal=1
# Wind
Weak=0, Strong=1

Labels
# Play:
No=0, Yes=1

```
conditions = [
    [0, 0, 0, 0],
    [0, 0, 0, 1],
    [1, 0, 0, 0],
    [2, 1, 0, 0],
    [2, 2, 1, 0],
    [2, 2, 1, 1],
    [1, 2, 1, 1],
    [0, 1, 0, 0],
    [0, 2, 1, 0],
    [2, 1, 1, 0],
    [0, 1, 1, 1],
    [1, 1, 0, 1],
    [1, 0, 1, 0],
    [2, 1, 0, 1]
]
```

```
labels = [
  0,
  0,
  1,
  1,
  1,
  0,
  1,
  0,
  1,
  1,
  1,
  1,
  1,
  0
]
```

```
data = { "data": conditions, "target": labels }

gnb = GaussianNB()
gnb = gnb.fit(data["data"], data["target"])

tdata = [0, 2, 0, 1]
prob_pred = gnb.predict_proba([tdata])
pred = gnb.predict([tdata])

print("Probability:", prob_pred)
print(pred)

# Probability: [[0.73580953 0.26419047]]
# [0]
```

Excursus - Naive Bayes Spam Filtering

ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive...
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives around here though
spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it...
ham	Even my brother is not like to speak with me. They treat me like aids patent.

https://archive.ics.uci.edu/ml/machine-learning-databases/00228/

Excursus - Naive Bayes Spam Filtering

ham Ok lar... Joking wif u oni...
spam Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive...
ham U dun say so early hor... U c already then say...
ham Nah I don't think he goes to usf, he lives around here though
spam FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it...
ham Even my brother is not like to speak with me. They treat me like aids patent.

ham = 0, spam = 1

https://archive.ics.uci.edu/ml/machine-learning-databases/00228/

Excursus - Naive Bayes Spam Filtering

label	message
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive...
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives around here though
spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it...
ham	Even my brother is not like to speak with me. They treat me like aids patent.

https://archive.ics.uci.edu/ml/machine-learning-databases/00228/

Excursus - Naive Bayes Spam Filtering

Hint

Excursus - Naive Bayes Spam Filtering

BoW - Bag of Words

(1) John likes to watch movies. Mary likes movies too.
(2) John also likes to watch football games.

"John","likes","to","watch","movies","Mary","likes","movies","too"
"John","also","likes","to","watch","football","games"

BoW1 = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1};
BoW2 = {"John":1,"also":1,"likes":1,"to":1,"watch":1,"football":1,"games":1};

https://en.wikipedia.org/wiki/Bag-of-words_model

Excursus - Naive Bayes Spam Filtering

CountVectorizer()

```
countVectorizer.fit(messages)
features = countVectorizer.get_feature_names()
vectorized_messages = countVectorizer.transform(messages).toarray()
```

	are	call	from	hello	home	how	me	money	now	tomorrow	win	you
0	1	0	0	1	0	1	0	0	0	0	0	1
1	0	0	1	0	1	0	0	1	0	0	2	0
2	0	1	0	0	0	0	1	0	1	0	0	0
3	0	1	0	2	0	0	0	0	0	1	0	1

https://towardsdatascience.com/naive-bayes-intuition-and-implementation-ac328f9c9718

Approach

- Clean and prepare the given data
- Label the data and store it
- Define the features you want to use
- Prepare your features / transform them into a format you can work with
- Train your model
- Evaluate your model
- Visualize your results

Sources

Sources - Python

- https://www.python.org/ (Basics & Documentation)
- https://app.finxter.com/learn/computer/science/ (Test your skills)
- https://en.wikibooks.org/wiki/Non-Programmer%27s_Tutorial_for_Python_3 (Basic Python Tutorial)
- https://thepythonguru.com/ (Basic / Advanced Python Tutorial)

Sources - ML / Naive Bayes

- http://guidetodatamining.com/ (A Programmer's Guide to Data Mining - free e book)
- https://www.dataquest.io/blog/sci-kit-learn-tutorial/ (Sci-Kit Learn Tutorial)
- https://www.analyticsvidhya.com/blog/2015/06/infographic-cheat-sheet-data-exploration-python/ (Cheat Sheet Data Analysis)
- https://www.analyticsvidhya.com/blog/2015/06/quick-guide-text-data-cleaning-python/ (Cheat Sheet Text Data Cleaning)
- https://scikit-learn.org/stable/modules/naive_bayes.html
- https://en.wikipedia.org/wiki/Bayes%27_theorem
- https://towardsdatascience.com/naive-bayes-intuition-and-implementation-ac328f9c9718 (Example for spam / ham classification)