Naive Bayes Classifier ¶

MSE TSM_MachLe, Christoph Würsch

Naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector $\mathbf{x} = (x_1, \dots, x_n)$ representing some n features (independent variables), it assigns to this instance probabilities

$$p(C_k \mid x_1, \dots, x_n)$$

for each of K possible outcomes or classes C_k

The problem with the above formulation is that if the number of features n is large or if a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable. Using **Bayes' theorem**, the conditional probability can be decomposed as:

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

Using Bayesian probability terminology, the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \cdot \text{likelihood}}{\text{evidence}}$$

In practice, there is interest only in the numerator of that fraction, because the denominator does not depend on C and the values of the features x_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model

$p(C_k, x_1, \dots, x_n)$ which can be rewritten as follows, using the chain rule for repeated applications of the definition of conditional probability:

$$\begin{aligned} p(C_k, x_1, \dots, x_n) &= p(x_1, \dots, x_n, C_k) \\ &= p(x_1 \mid x_2, \dots, x_n, C_k)\ p(x_2, \dots, x_n, C_k) \\ &= p(x_1 \mid x_2, \dots, x_n, C_k)\ p(x_2 \mid x_3, \dots, x_n, C_k)\ p(x_3, \dots, x_n, C_k) \\ &= \cdots \\ &= p(x_1 \mid x_2, \dots, x_n, C_k)\ p(x_2 \mid x_3, \dots, x_n, C_k) \cdots p(x_{n-1} \mid x_n, C_k)\ p(x_n \mid C_k)\ p(C_k) \end{aligned}$$

Now the "naive" conditional independence assumptions come into play: assume that all features in $\mathbf{x}$ are mutually independent, conditional on the category C_k. Under this assumption,

$$p(x_i \mid x_{i+1}, \dots, x_n, C_k) = p(x_i \mid C_k)$$

Thus, the joint model can be expressed as

$$\begin{aligned} p(C_k \mid x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &= p(C_k)\ p(x_1 \mid C_k)\ p(x_2 \mid C_k)\ p(x_3 \mid C_k)\ \cdots \\ &= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,, \end{aligned}$$

where $\propto$ denotes proportionality.

This means that **under the above independence assumptions**, the conditional distribution over the class variable C is:

$$p(C_k \mid x_1, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

where the evidence

$$Z = p(\mathbf{x}) = \sum_k p(C_k)\ p(\mathbf{x} \mid C_k)$$

is a scaling factor dependent only on $x_1, \dots, x_n$ that is, a constant if the values of the feature variables are known.

Source: [https://en.wikipedia.org/wiki/Naive_Bayes_classifier (https://en.wikipedia.org/wiki/Naive_Bayes_classifier)](https://en.wikipedia.org/wiki/Naive_Bayes_classifier)

An introductory example: Mrs Marple plays Golf

Let's analyze the probabilty for Mrs Marple to play Golf on a given day. Mrs Marple has recorded her decisions on 14 different days. Depending on the `Temperature`, the `Outlook`, the `Humidity` and `Wind Condition` of the day, we want to predict, whether she is going to play Golf or not.

- X_1: `Temperature`
- X_2: `Outlook`
- X_3: `Humidity`
- X_4: `Windy?`

- $\text{dom}(X_1) = \{\text{hot, cold, mild}\}$
- $\text{dom}(X_2) = \{\text{sunny, overcast, rain}\}$
- $\text{dom}(X_3) = \{\text{normal, high}\}$
- $\text{dom}(X_4) = \{\text{True, False}\}$

The response y is: `Play Golf?`.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

df=pd.read_excel('PlayGolf_NaiveBayes.xlsx')
df.head(20)
```

Out[1]:

	Temperature	Outlook	Humidity	Windy	Play Golf?
0	hot	sunny	high	False	no
1	hot	sunny	high	True	no
2	hot	overcast	high	False	yes
3	cool	rain	normal	False	yes
4	cool	overcast	normal	True	yes
5	mild	sunny	high	False	no
6	cool	sunny	normal	False	yes
7	mild	rain	normal	False	yes
8	mild	sunny	normal	True	yes
9	mild	overcast	high	True	yes
10	hot	overcast	normal	False	yes
11	mild	rain	high	True	no
12	cool	rain	normal	True	no
13	mild	rain	high	False	yes

1. Calculation of the Prior

The prior probabilty is the probability to play Golf at all. For this we only have to look at the last column.

$$p(y = \text{yes}) = \frac{9}{14}$$
$$p(y = \text{no}) = \frac{5}{14}$$

2. Calculation of the conditional probabilities $p(y|X_i)$

As a next step, we have to calculate all conditional probabilities, i.e. the probability for every feature X_i given $y = \text{yes}$ and $y = \text{no}$. We can use a contingency table (`pandas.cosstab`) to calculate these probabilities.

In [2]:
```
pd.crosstab(df['Temperature'],df['Play Golf?'],margins=True,normalize=False)
```

Out[2]:

Play Golf?	no	yes	All
Temperature			
cool	1	3	4
hot	2	2	4
mild	2	4	6
All	5	9	14

From this table, we can calculate the following:

- $p(X_1 = \text{cool}|y = \text{yes}) = \frac{3}{9}$
- $p(X_1 = \text{hot }|y = \text{yes}) = \frac{2}{9}$
- $p(X_1 = \text{mild}|y = \text{yes}) = \frac{4}{9}$
- $p(X_1 = \text{cool}|y = \text{no }) = \frac{1}{5}$
- $p(X_1 = \text{hot }|y = \text{no }) = \frac{2}{5}$
- $p(X_1 = \text{mild}|y = \text{no }) = \frac{2}{5}$

In [3]:
```
pd.crosstab(df['Outlook'],df['Play Golf?'],margins=True)
```

Out[3]:

Play Golf?	no	yes	All
Outlook			
overcast	0	4	4
rain	2	3	5
sunny	3	2	5
All	5	9	14

From this table, we can calculate the following conditionals:

- $p(X_2 = \text{overcast}|y = \text{yes}) = \frac{4}{9}$
- $p(X_2 = \text{rain }|y = \text{yes}) = \frac{3}{9}$
- $p(X_2 = \text{sunny}|y = \text{yes}) = \frac{2}{9}$
- $p(X_2 = \text{overcast}|y = \text{no }) = \frac{0}{5}$
- $p(X_2 = \text{rain }|y = \text{no }) = \frac{2}{5}$
- $p(X_2 = \text{sunny}|y = \text{no }) = \frac{3}{5}$

In [4]:
```
pd.crosstab(df['Humidity'],df['Play Golf?'],margins=True)
```

Out[4]:

Play Golf?	no	yes	All
Humidity			
high	4	3	7
normal	1	6	7
All	5	9	14

From this table, we can calculate the following conditionals:

- $p(X_3 = \text{high }|y = \text{yes}) = \frac{3}{9}$
- $p(X_3 = \text{normal}|y = \text{yes}) = \frac{6}{9}$
- $p(X_3 = \text{high }|y = \text{no }) = \frac{4}{5}$
- $p(X_3 = \text{normal}|y = \text{no }) = \frac{1}{5}$

In [5]:
```
pd.crosstab(df['Windy'],df['Play Golf?'],margins=True)
```

Out[5]:

Play Golf?	no	yes	All
Windy			
False	2	6	8
True	3	3	6
All	5	9	14

And finally for the last feature X_4, we get

- $p(X_4 = \text{False }|y = \text{yes}) = \frac{6}{9}$
- $p(X_4 = \text{True}|y = \text{yes}) = \frac{3}{9}$
- $p(X_4 = \text{False }|y = \text{no }) = \frac{2}{5}$
- $p(X_4 = \text{True}|y = \text{no }) = \frac{3}{5}$

3. Making predictions

What is now the decision to play Golf under the the following conditions?

$$X = \{X_1, X_2, X_3, X_4\} = \{\text{hot, sunny, high, True}\}$$

It is not necessary, to calculate the evidence $Z = p(X)$, because we can compare the following two nominators:

$$p(y = \text{yes}|X) \propto p(X_1|y) \cdot p(X_2|y) \cdot p(X_3|y) \cdot (X_4|y) \cdot p(y)$$

and

$$p(y = \text{no}|X) \propto p(X_1|\bar{y}) \cdot p(X_2|\bar{y}) \cdot p(X_3|\bar{y}) \cdot (X_4|y) \cdot p(\bar{y})$$

By looking at our tables, we get for the probability to play Golf $y = \text{yes}$:

$$p(y = \text{yes}|X) = \frac{1}{Z} \cdot p(X_1|y) \cdot p(X_2|y) \cdot p(X_3|y) \cdot (X_4|y) \cdot p(y) = \frac{2}{9} \cdot \frac{2}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} \cdot \frac{9}{14}$$
$$= \frac{2^2}{9^2} \cdot \frac{1}{14Z}$$

By looking at our tables, we get for the probability **not** to play Golf: $y = \text{no}$:

$$p(y = \text{not}|X) = \frac{1}{Z} \cdot p(X_1|\bar{y}) \cdot p(X_2|\bar{y}) \cdot p(X_3|\bar{y}) \cdot (X_4|\bar{y}) \cdot p(\bar{y}) = \frac{2}{5} \cdot \frac{2}{5} \cdot \frac{4}{5} \cdot \frac{3}{5} \cdot \frac{5}{14}$$
$$= \frac{2^4}{5^3} \cdot \frac{1}{14Z}$$

Now, we can compare these two probabilites and calculate the *ratio*:

$$\frac{p(y = \text{yes}|X = \{\text{hot, sunny, high, True}\})}{p(y = \text{no}|X = \{\text{hot, sunny, high, True}\})} = \frac{2^2 \cdot 5^3}{9^2 \cdot 2^4} = \frac{125}{4 \cdot 81} = \frac{125}{324} < 1$$

4. What happens, if one of the counts is zero?

In this case, the posterior $p(y|X)$ is zero as well. To avoid this, **Additive Smoothing (Laplace Smoothing)** is normally applied to the data.

$$p_i = \frac{n_i + 1}{n + k}$$

where:

- n_i: is the actual count of characteristic i of the feature X, $(i = 1 \dots k)$ conditioned on y
- k: is the number of classes (characteristics) in the considered feature X conditioned on y
- n: is the total number of counts of this feature conditioned on y

Interested readers can have a look at: https://en.wikipedia.org/wiki/Additive_smoothing (https://en.wikipedia.org/wiki/Additive_smoothing)

```
In [ ]:
```