

# blatt08\_Guth\_Venker\_Jaekel

June 22, 2021

## 1 Abgabe SMD Blatt 08

### 1.0.1 von Nico Guth, David Venker, Jan Jäkel

#### 1.1 Aufgabe 16

Weiter unten im PDF

#### 1.2 Aufgabe 17

##### 1.2.1 a) und b)

Weiter unten im PDF

##### 1.2.2 c)

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

```
[2]: import matplotlib as mpl  
mpl.rcParams['font.size'] = 15
```

```
[3]: # Datensatz aus der Aufgabe  
df = pd.DataFrame(  
    [[29.4, 2, 85, False, False],  
     [26.7, 2, 90, True, False],  
     [28.3, 1, 78, False, True],  
     [21.1, 0, 96, False, True],  
     [20.0, 0, 80, False, True],  
     [18.3, 0, 70, True, False],  
     [17.8, 1, 65, True, True],  
     [22.2, 2, 95, False, False],  
     [20.6, 2, 70, False, True],  
     [23.9, 0, 80, False, True],  
     [23.9, 2, 70, True, True],  
     [22.2, 1, 90, True, True],  
     [27.2, 1, 75, False, True],  
     [21.7, 0, 80, True, False]],
```

```

        columns=['Temperatur', 'Wettervorhersage', 'Luftfeuchtigkeit', u
        →'Wind', 'Fussball']
    )

```

## Funktionsdefinitionen

```
[4]: # calculate the probability that the condition occurs
# given that the conditional_condition occurs
# with masks of the dataset as parameters
def P(condition,conditional_condition=None):
    if conditional_condition is None:
        return condition.sum() / condition.size
    else:
        if np.any(conditional_condition):
            return (condition&conditional_condition).sum() / u
        →conditional_condition.sum()
        else:
            return 0
```

```
[5]: def xlog2x(P):
    if P == 0:
        return 0
    else:
        return P*np.log2(P)
```

```
[6]: # calculate the entropy of the X<=cut part of the dataset
def H_low(Y,X,cut):
    return -1 * ( xlog2x(P(Y==True, X<=cut)) + xlog2x(P(Y==False, X<=cut)) )

# calculate the entropy of the X>cut part of the dataset
def H_high(Y,X,cut):
    return -1 * ( xlog2x(P(Y==True, X>cut)) + xlog2x(P(Y==False, X>cut)) )
```

```
[7]: # calculate the information gain when cutting at cut in the given feature X
def IG(Y,X,cut):
    H_after = P(X<=cut) * H_low(Y,X,cut) + P(X>cut) * H_high(Y,X,cut)
    return H_before - H_after
```

## Entropie der Wurzel berechnen

```
[8]: Y = df['Fussball']
```

```
[9]: H_before = - xlog2x(P(Y==True)) - xlog2x(P(Y==False))
print(f'Entropy before the decision: {H_before}')
```

Entropy before the decision: 0.9402859586706311

## Plot des Informationsgewinns pro Attribut je nach Schnitt

```
[10]: X_cols = ['Temperatur', 'Wettervorhersage', 'Luftfeuchtigkeit', 'Wind']
```

```
[11]: for X_col in X_cols:
    # calculate all the information gains of a particular feature
    X = df[X_col]
    dX = 1
    cuts = np.linspace(X.min()-dX,X.max()+dX,500)
    IGs = np.zeros_like(cuts)
    for i,cut in enumerate(cuts):
        IGs[i] = IG(Y,X,cut)

    # plot the IG to the cuts
    plt.figure(figsize=(6,6))
    plt.title(X_col)
    plt.plot(cuts,IGs)
    plt.xlabel('Cut')
    plt.ylabel('IG')
    plt.show()

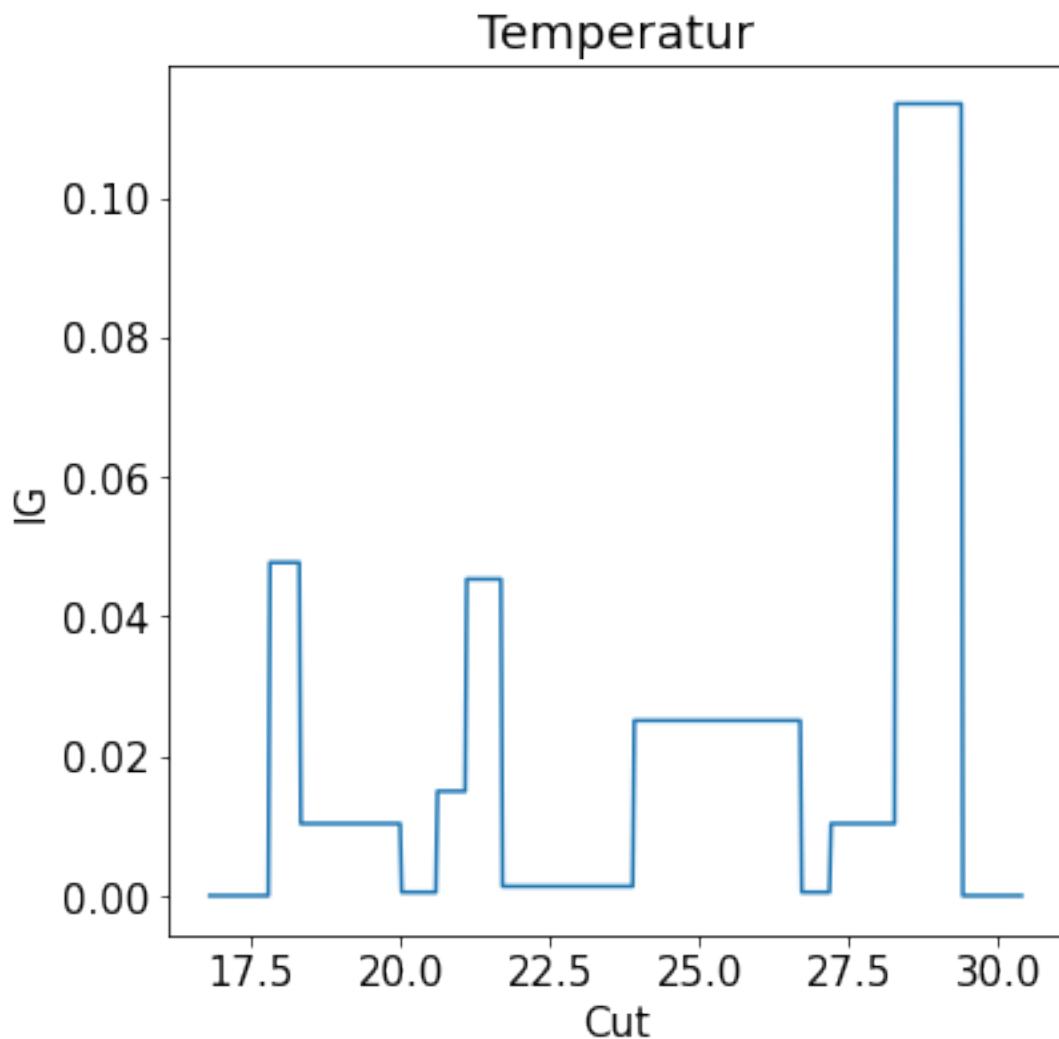
    # print the IG values
    with np.printoptions(precision=5):
        print(f'IG Werte: {np.unique(IGs)}')

    # print the best IG
    IG_max = np.max(np.unique(IGs))
    print(f'Maximaler IG: {IG_max:.8f}')

    # print the cut interval of the best IG
    IG_max_indices = np.where(IGs == IG_max)[0]
    min_cut = cuts[IG_max_indices[0]]
    max_cut = cuts[IG_max_indices[-1]]
    print(f'Bestes Schnittintervall: ({min_cut:.1f},{max_cut:.1f})')

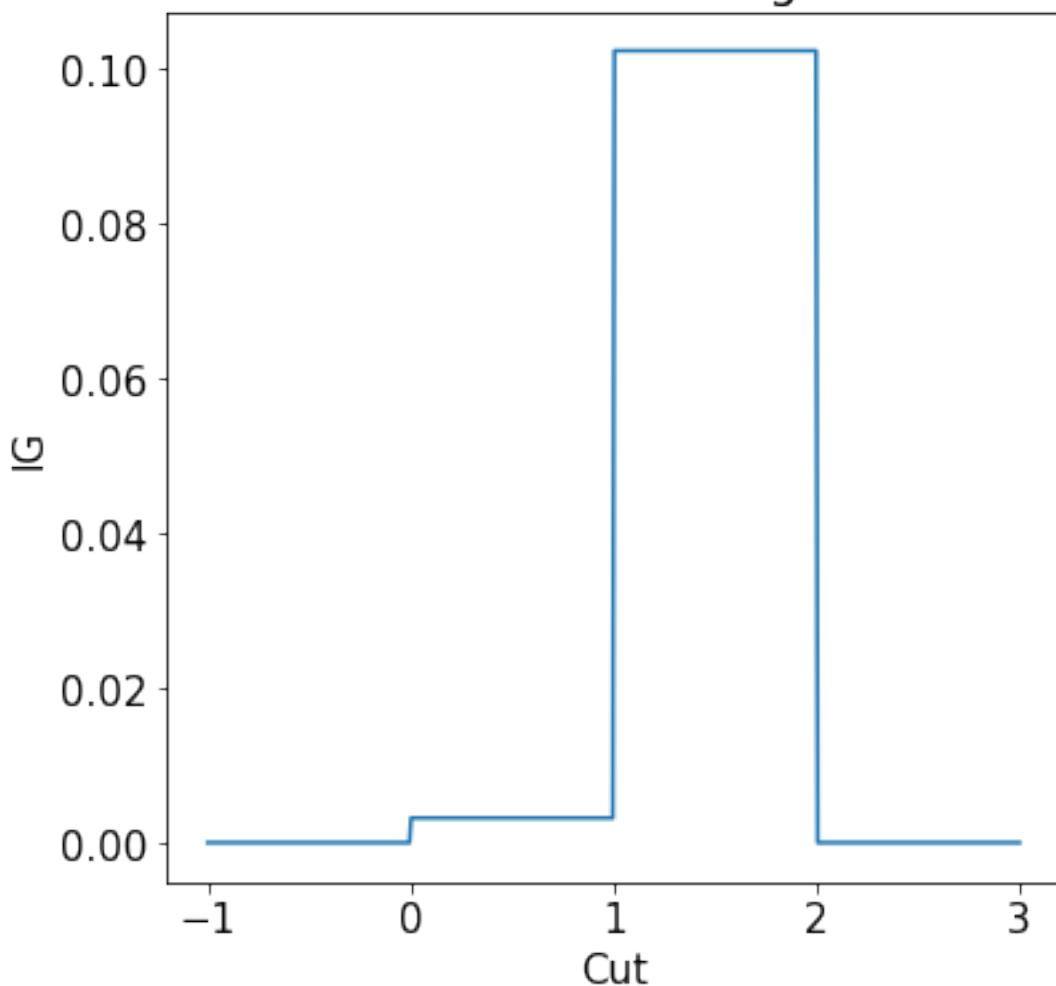
    # print size of the target classes if the data was cut at the highest IG
    print(f'Y(X<=Cut): (True: {Y[X<=min_cut].sum()} ; False: {(~Y[X<=min_cut]).sum()})')
    print(f'Y(X>Cut) : (True: {Y[X>min_cut].sum()} ; False: {(~Y[X>min_cut]).sum()})')

    print('\n')
```



```
IG Werte: [0.          0.00049  0.00134  0.01032  0.01496  0.02508  0.04533  0.04771
0.1134 ]  
Maximaler IG: 0.11340086  
Bestes Schnittintervall: (28.3,29.4)  
Y(X<=Cut): (True: 9 ; False: 4)  
Y(X>Cut) : (True: 0 ; False: 1)
```

## Wettervorhersage



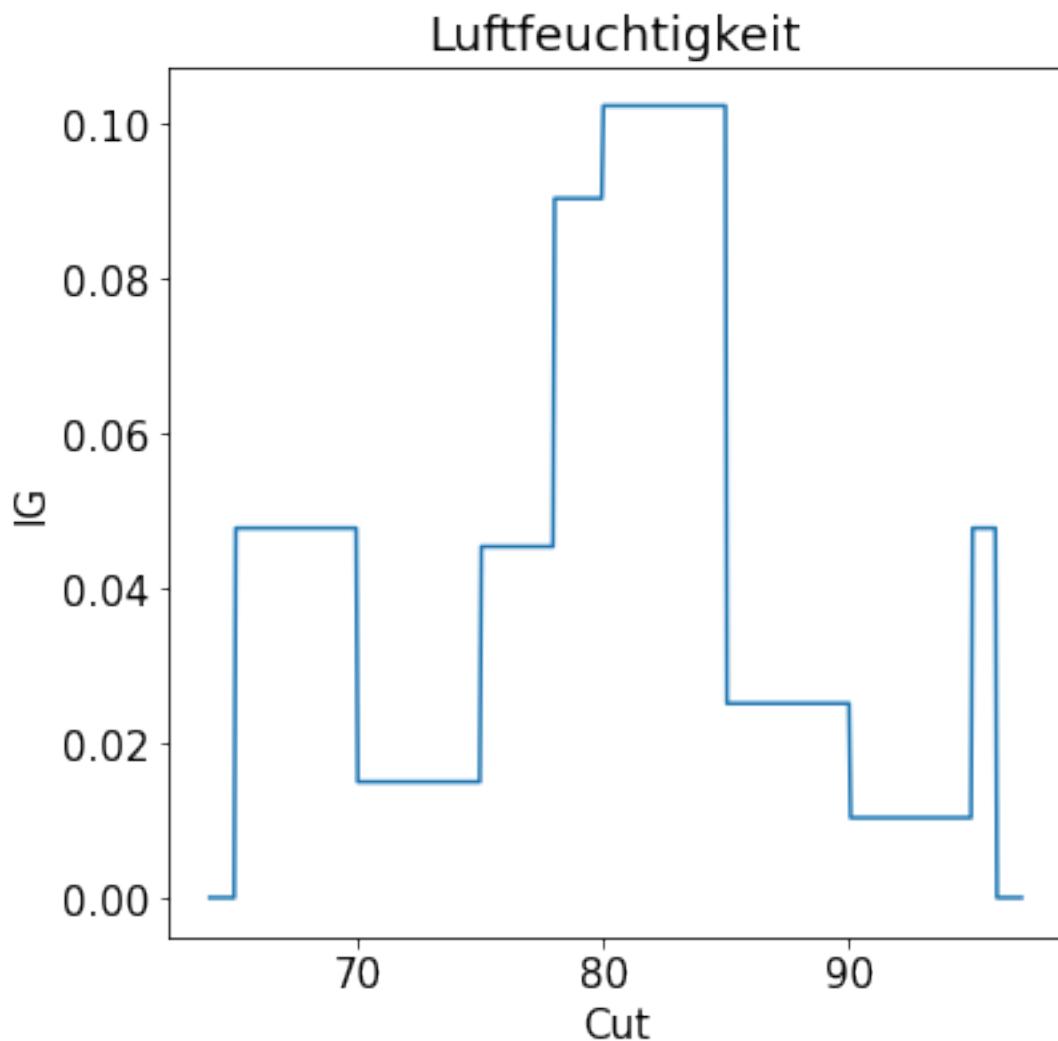
IG Werte: [0. 0.00318 0.10224]

Maximaler IG: 0.10224356

Bestes Schnittintervall: (1.0,2.0)

Y(X<=Cut): (True: 7 ; False: 2)

Y(X>Cut) : (True: 2 ; False: 3)



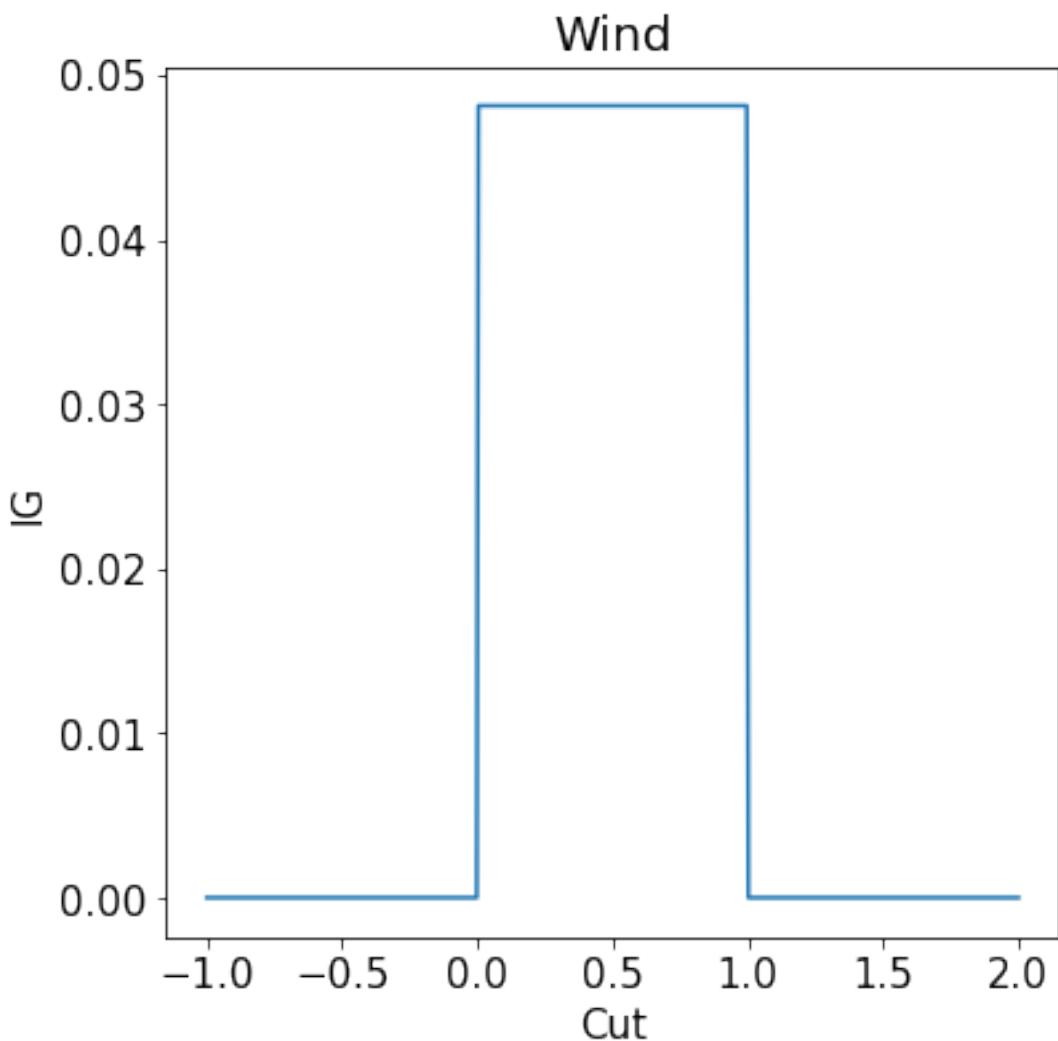
IG Werte: [0. 0.01032 0.01496 0.02508 0.04533 0.04771 0.09028 0.10224]

Maximaler IG: 0.10224356

Bestes Schnittintervall: (80.0,85.0)

Y(X<=Cut): (True: 7 ; False: 2)

Y(X>Cut) : (True: 2 ; False: 3)



```

IG Werte: [0.      0.04813]
Maximaler IG: 0.04812703
Bestes Schnittintervall: (0.0,1.0)
Y(X<=Cut): (True: 6 ; False: 2)
Y(X>Cut) : (True: 3 ; False: 3)

```

### 1.2.3 d) Welches Attribut eignet sich am besten zum Trennen der Daten?

Den höchsten Informationsgewinn hat der Schnitt in der Temperatur.

Aber hier wird nur ein Wert abgeschnitten und somit sollte nicht die Temperatur als erstes verwendet werden.

Knapp dahinter sind die Informationsgewinne bei Schnitt in der Wettervorhersage oder der Luftfeuchtigkeit.

Beide Attribute schneiden den Datensatz genau gleich.

Allerdings scheint die Verteilung der IGs der Luftfeuchtigkeit annähernd einer Normalverteilung zu folgen.

Also würden wir uns dazu entscheiden in der Luftfeuchtigkeit die erste Entscheidung zu setzen.

[ ]:

# Naive Bayes: Fußball

Freitag, 18. Juni 2021 11:57

$$\text{Satz von Bayes: } P(F|w) = \frac{P(w|F) \cdot P(F)}{P(w)}$$

$$a) P(A|B) = \frac{P(A \cap B)}{P(B)} \quad \text{mit} \quad P(A \cap B) = P(A) \cdot P(B|A)$$

$$\text{Bd. V. } P(F \cap w) = P(F) \cdot P(w|F)$$

$$P(w \cap F) = P(w) \cdot P(F|w)$$

$$\Leftrightarrow P(F|w) = \frac{P(w|F) \cdot P(F)}{P(w)}$$

$$b) \text{ gesucht: } P(F|w)$$

$$P(w|F) = \prod_i P(x_i|F) = \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{2}{9} = \frac{2}{243}$$

$$P(F)_{\text{A.Priori.}} = \frac{1}{2}$$

$$\begin{aligned} P(w)_{\text{A.Priori.}} &= P(w|F=\text{Ja}) \cdot P(F=\text{Ja}) + P(w|F=\text{Nein}) \cdot P(F=\text{Nein}) \\ &= \frac{54}{6567} \cdot \frac{1}{2} + \frac{36}{625} \cdot \frac{1}{2} = 0,033 \end{aligned}$$

$$P(F|w) = \frac{P(w|F) \cdot P(F)}{P(w)} = \frac{\frac{2}{243} \cdot \frac{1}{2}}{0,033} = 0,129$$

$$c) P(w|F) = 0, \text{ da } P(\text{heiß} | F) = 0$$

Der Datensatz ist nicht aussagekräftig genug.

$$P(F) = \frac{1}{2}$$

$$P(W) = P(W | F=Ja) \cdot P(F=Ja) + P(W | F=Nein) \cdot P(F=Nein)$$

Idee: Wir entfernen Temperatur als Attribut.

$$P(W) = \frac{36}{729} \cdot \frac{1}{2} + \frac{8}{125} \cdot \frac{1}{2} = 0,057$$

$$P(W|F) = \frac{6}{5} \cdot \frac{3}{5} \cdot \frac{2}{9} = \frac{2}{5} \cdot \frac{1}{3} \cdot \frac{2}{9} = \frac{4}{81}$$

$$P(F|W) = \frac{\frac{4}{81}}{0,057} = 0,633$$

Das Attribut Temperatur nur in "heiß" etc einzuteilen ist nicht sehr aussagekräftig.

## Aufgabe 17:

Tabelle 4: Datensatz: „Soll ich Fußballspielen gehen?“

Temperatur / °C	Wettervorhersage	Luftfeuchtigkeit / %	Wind	Fußball
29,4	2	85	False	False
26,7	2	90	True	False
28,3	1	78	False	True
21,1	0	96	False	True
20	0	80	False	True
18,3	0	70	True	False
17,8	1	65	True	True
22,2	2	95	False	False
20,6	2	70	False	True
23,9	0	80	False	True
23,9	2	70	True	True
22,2	1	90	True	True
27,2	1	75	False	True
21,7	0	80	True	False

a) Entropie der Wurzel:

$$\text{Entropie: } H(Y) = - \sum_{z \in Z} P(Y=z) \log_2 (P(Y=z))$$

hier  $Z = \{\text{True}, \text{False}\} = \{F, T\} = \{\text{Fußball}, \text{nicht Fußball}\}$

$Y \triangleq \text{Fußball}$

$$H(Y) = - \frac{n_T}{n} \cdot \log_2 \frac{n_T}{n} - \frac{n_F}{n} \log_2 \frac{n_F}{n}$$

$$n=14$$

$$n_T = 9 \quad n_F = 5$$

$$H(Y) = - \frac{9}{14} \cdot (\log \frac{9}{14}) - \frac{5}{14} \cdot \log_2 \frac{5}{14}$$

$$= 0.940286 \dots$$

b)

Informationsgewinn bei Schnitt auf Attribut Wind

Informationsgewinn:

$$IG(X, Y) = H(Y) - H(Y|X)$$

$$H(Y) = - \sum_{z \in Z} P(Y=z) \log_2 (P(Y=z))$$

$$H(Y|X) = \sum_{m \in M} P(X=m) H(Y|X=m)$$

$$= - \sum_{m \in M} (P(X=m) \cdot \sum_{z \in Z} P(Y=z | X=m) \log_2 (P(Y=z | X=m)))$$

hier (Wind):  $M = \{W, \neg W\} \quad Z = \{F, \neg F\}$

hier ( $X_{\text{Wind}}$ ):  $M = \{W, \neg W\}$      $Z = \{F, \neg F\}$

$$H(Y|X_{\text{Wind}}) = -P(X=W) \cdot \left( P(Y=F|X=W) \log_2 \left( P(Y=F|X=W) + P(Y=\neg F|X=W) \log_2 \left( P(Y=\neg F|X=W) \right) \right) \right. \\ \left. - P(X=\neg W) \cdot \left( P(Y=F|X=\neg W) \log_2 \left( P(Y=F|X=\neg W) + P(Y=\neg F|X=\neg W) \log_2 \left( P(Y=\neg F|X=\neg W) \right) \right) \right) \right)$$

$$H(Y|X_{\text{Wind}}) = - \frac{n(W)}{n} \cdot \left( \frac{n(F \wedge W)}{n(W)} \cdot \log_2 \left( \frac{n(F \wedge W)}{n(W)} \right) + \frac{n(\neg F \wedge W)}{n(W)} \log_2 \left( \frac{n(\neg F \wedge W)}{n(W)} \right) \right) \\ - \frac{n(\neg W)}{n} \cdot \left( \frac{n(F \wedge \neg W)}{n(\neg W)} \log_2 \left( \frac{n(F \wedge \neg W)}{n(\neg W)} \right) + \frac{n(\neg F \wedge \neg W)}{n(\neg W)} \log_2 \left( \frac{n(\neg F \wedge \neg W)}{n(\neg W)} \right) \right)$$

$$n = 14$$

$$n(W) = 6 \quad n(\neg W) = 8$$

$$n(F \wedge W) = 3 \quad n(\neg F \wedge W) = 3$$

$$n(F \wedge \neg W) = 6 \quad n(\neg F \wedge \neg W) = 2$$

$$H(Y|X_{\text{Wind}}) = - \frac{1}{14} \left( 3 \log_2 \left( \frac{3}{6} \right) + 3 \log_2 \left( \frac{3}{6} \right) \right. \\ \left. - \frac{1}{14} \left( 6 \cdot \log_2 \left( \frac{6}{8} \right) + 2 \log_2 \left( \frac{2}{8} \right) \right) \right) \\ = 0.892159$$

$$G(Y|X_{\text{Wind}}) = H(Y) - H(Y|X_{\text{Wind}})$$

$$= 0.940286 - 0.892159$$

$$= 0.048127$$