UNIVERSITÉ DE LIÈGE

[ELEN0060] INFORMATION AND CODING THEORY

Project 1 - Information measures

Authors:
MENDIZABAL Emma (s218926)& BERNARD Aurélien (s176639)

March 27, 2022

1 Implementation

1.1 Entropy

Given a random variable X and a its probability distribution $P_X = (p_1, p_2, ..., p_n)$ the entropy of the variable can be computed by equation 1 here bellow.

$$H(X) \triangleq -\sum_{i=1}^{n} p_i \log(p_i) \tag{1}$$

In order to implement this function a simple loop over the given probability was used. The summed term was computed at each iteration and added to the previous ones using an accumulator variable.

Intuitively, entropy is a measure of uncertainty. The higher the entropy the less is known from a system. High entropy can be found in systems where there is a high number of possible outcomes and uniform probability distribution between all the outcomes

1.2 Joint entropy

Given the joint probability distribution of two random variables X and Y as the matrix shown in equation 2 here bellow, the joint entropy can be computed using equation 3.

$$P_{X,Y} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ p_{n1} & p_{n2} & \cdots & p_{nm} \end{bmatrix} \text{Where } p_{ij} = P(X_i \cap Y_j) \tag{2}$$

$$H(X,Y) \triangleq -\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij} \log(p_{ij}) \tag{3}$$

To implement joint entropy a very similar approach to the one used previously for entropy was used. The main difference being that this time, 2 nested loops were used to iterate over every element of the array. As for the two implementations, the equation for entropy and joint entropy are very similar. The only difference being the fact that the entropy is computed for a joint entropy so now 2 summations are used in the equation to sum every combination of joint probabilities.

1.3 Conditional entropy

As in the previous section a joint probability distribution was given. However, it was asked to implement the conditional entropy $H(X \mid Y)$. To do so equation 4 corresponding to the chain rule of conditional entropy was used.

$$H(X \mid Y) = H(X,Y) - H(Y) \tag{4}$$

In order to implement this equation the marginal probability distribution for variable Y was computed. The marginal distribution was computed with a loop by adding all values of p_{ij} for a given i. Mathematically, the following equation was used: $P_Y = \sum_{i=1}^{n} p_{ij}$. After computing the marginal probability distribution, the entropy and joint entropy functions implemented for the previous sections were used to compute $H(X,Y)$ and $H(Y)$. Finally these two values were subtracted and returned.

An alternative to this implementation would be to directly apply the first equation given in the theoretical course to compute conditional entropy. This equation can be seen in equation 5. The equation that we used can be obtained from this equation by applying logarithm identities and expanding the obtained equation. The full derivation can be found in equation 6

$$
\begin{gathered}
H(X \mid Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(p_{ij}) \log P\left(X_i \mid Y_j\right), \\
P(X_i \mid Y_j) = \frac{p_{ij}}{\sum_{j=1}^{m} p_{ij}}
\end{gathered} \tag{5}
$$

$$
\begin{aligned}
\mathrm{H}(Y \mid X) &= \sum_{x\in X, y\in Y} p(x,y) \log\left(\frac{p(x)}{p(x,y)}\right) \\
&= \sum_{x\in X, y\in Y} p(x,y)(\log(p(x)) - \log(p(x,y))) \\
&= -\sum_{x\in X, y\in Y} p(x,y)\log(p(x,y)) + \sum_{x\in X, y\in Y} p(x,y)\log(p(x)) \\
&= \mathrm{H}(X,Y) + \sum_{x\in X} p(x)\log(p(x)) \\
&= \mathrm{H}(X,Y) - \mathrm{H}(X).
\end{aligned} \tag{6}
$$

1.4 Mutual information

Using a joint probability distribution, equation 7 was applied to compute mutual information.

$$
I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{7}
$$

To implement equation 7 we mainly used the functions for entropy and joint entropy that were defined for the previous sections. From this formula we can see that mutual information is symmetric with respect to its variables. Meaning that $I(X;Y) = I(Y;X)$. It can also be note that mutual information cannot be negative. Using the conditional entropy formula used in the previous section we can also express mutual information as $I(X;Y) = H(X) + H(Y) - H(Y) - H(X|Y)$ and we know that $H(X) \geq H(X|Y)$ as in the worst case, Y does not bring information $\implies P(X) \equiv P(X|Y) \Leftrightarrow H(X) \equiv H(X|Y)$.

1.5 Adding a third random variable

Conditional joint entropy $H(X,Y|Z)$ was computed using the following formula.

$$
H(X,Y|Z) = H(X|Z) + H(X,Y,Z) - H(X,Z) \tag{8}
$$

In order to compute the conditional joint entropy the joint entropy of 3 variables had to be computed. The joint entropy for three variables was computed by looping over every joint probability and applying the previously seen entropy function. Then the joint probability $P_{X,Z}$ was computed by summing all values across the Y axis and the formula was applied using previously implemented functions to finally get the conditional joint entropy.

Conditional mutual information $I(X;Y|Z)$ was computed using the following equations.

$$
I(X;Y|Z) = H(X|Z) - H(X|Y,Z), \text{ with } H(X|Y,Z) = H(X,Y,Z) - H(Y,Z) \tag{9}
$$

The equation was implemented using previously implemented functions and by summing probabilities to get the required marginal probability distributions.

2 Weather forecasting

2.1 Entropy and cardinality

For this second part, we used the "weather_data.csv" file provided. This database is made of 14 columns, each describing a feature of weather (air pressure, temperature, etc ...), and 5000 rows describing weather at a specific time.

Thanks to the functions implemented in part 1, we first computed the entropy of each feature for these 5000 cases. The results are shown in table 1 here below :

feature	corresponding entropy	cardinality of values
'temperature'	1.51	3
'air_pressure'	0.99	2
'same_day_rain'	1.47	3
'next_day_rain'	1.56	3
'relative_humidity'	0.99	2
'wind_direction'	1.99	4
'wind_speed'	1.58	3
'cloud_height'	1.58	3
'cloud_density'	1.58	3
'month'	3.58	12
'day'	2.80	7
'daylight'	0.99	2
'lightning'	0.32	3
'air_quality'	0.53	3

Table 1: Table showing each feature of the *weather_data.csv* file alongside their respective entropy and cardinality.

We can notice that the entropy and cardinality of features are linked to the relative distribution uniformity. They respect the following formula:

$$\text{For a uniform distribution, } 2^{entropy} = cardinality \tag{10}$$

$$\text{For a non-uniform distribution, } 2^{entropy} < cardinality \tag{11}$$

Hence we can notice that for equal cardinalities, features with higher entropies are closer to a uniform distribution than those with lower entropies. For instance, the "wind_speed" distribution is closer to uniformity than the "lightening" one.

2.2 Conditional entropy

We computed the entropy of the feature "next_day_rain", given each one of the other features. The results are shown in table 2 here below :

entropy of next_day_rain given (...)	value
temperature	3.07
air_pressure	1.93
same_day_rain	2.86
next_day_rain	1.56
relative_humidity	2.30
wind_direction	3.56
wind_speed	3.15
cloud_height	3.15
cloud_density	3.15
month	5.14
day	4.37
daylight	2.56
lightning	1.89
air_quality	2.10

Table 2: Entropy value of the feature ”next_day_rain” given the each one of the other features.

The conditional entropy of next_day_rain given wind_direction is higher than the one given same_day_rain. Hence, it seems that values of next_day_rain depend more on those of same_day_rain than those of wind_direction.

2.3 Mutual information

We computed the mutual information between different features. The results we want to focus on are here below :

- mutual information between relative_humidity and wind_speed : 0.12e-03

  The result is close to zero: we can deduce that relative_humidity and wind_speed are not particularly linked.

- mutual information between month and temperature : 0.57

  Here the mutual information is higher: the temperature and the month have a stronger link. That result can be validated intuitively (in areas with temperate climate at least).

2.4 Which feature

In this part, we want to improve next_day_rain forecasting while being restricted to knowing one other feature only. We used mutual information, then conditional entropy to find this feature.

We first computed the mutual information between next_day_rain and each other feature, to find the one that maximises it. We found ”air_pressure” was the best feature to do so.

Then we computed the conditional entropy of next_day_rain knowing each other feature,to find the one that minimises it. Once again, we found ”air_pressure” was the best feature to do so. Thus the choice of the tool (conditional entropy or mutual information) is not important here.

2.5 Removing ”dry” days

We generated a new database by removing all rows with a ”dry” next_day_rain value. Then we reiterated the actions of part 2.4: this time, ”relative_humidity” seems to be the best feature to measure. This result can be validated intuitively.

2.6 Conditional mutual information

In this part, we have the opportunity to know both temperature and an other feature in order to forecast next_day_rain. Our role once again is to find the best feature to do so.

We computed the conditional mutual information between next_day_rain, temperature and each other feature. We found the maximum was reached for the ”month” feature. Thus knowing the temperature, we would change our answer and collect the ”month” information instead of the ”relative_humidity”.

3 Wordle : Playing with information based strategy

3.1 Initial Entropy of simplified game

Given that there are 26 letters in the English alphabet and in the simple version of wordle we consider them equiprobable, the Shannon entropy of each field is 4.7004 bits. As the value in each field is independent from each other we can multiply by 5 the entropy of a single field to get the value for the entropy of the word. The total entropy of the game is 23.502 bits.

3.2 After the first guess

After the first guess one field has been solved. Its entropy is therefore 0. This reduces the entropy of the word as one less value has to be found. We also know that the other 4 letters are not part of the word. Thus, reducing the number of possibilities for each field to 22. The values in these fields are still independent from each other. Therefore, the remaining entropy is equal to $4 \cdot \log_2(22) = 17.837$bits. The guess brought $23.502 - 17.837 = 5.665$ bits of information.

3.3 Second guess

With this second guess we find 4 letters that are not in the word and a letter that is in the word but not in the correct place. Therefore, for all remaining positions we reduce the number of possible letters by 4. Except for the 4th field where we reduce it by 5. This is the field where the letter with the incorrect position was found. We also reduce the number of possible words to words where the letter ’G’ is present at least once.

The analytical development of the entropy of the word being harder, this time an empirical approach was used[1]. We simulated the game up until the second guess and enumerated the remaining possible words. There are 15623 different remaining words. If considered equiprobable this yields an entropy of 13.93 bits. Enumeration was also used to compute the entropy of each cell. The entropies where the following:

- Cell 1: 3.58 bits
- Cell 2: 0 bits
- Cell 3: 3.58 bits
- Cell 4: 4.08 bits
- Cell 5: 3.58 bits

[1] The code used in order to preform the empirical analysis is available at the end of the submitted jupyter notebook.

If the entropies are summed they add up to 14.82 bits. This value is higher than the value found by enumerating the words. This is because after a yellow letter is guessed a dependence between the cells arises. Therefore, the entropies cannot be summed in order to get the total entropy of the game.

3.4 Entropy of real game

The entropy of the real game would be much lower as there is a smaller number of possible words by only considering English words of 5 letters within a 2000 words dictionary. The entropy is reduced to 10.96 bits if we assume every word in the 2000 words dictionary is equiprobable.

3.5 Information based strategy

A good strategy would be to reduce as much as possible the entropy of the system with each guess. This should reduce the possibilities enough to end up with only one option. In order to know which word would reduce the most the entropy of the game at each guess we should compute the expectation of the information that the guess would bring. To do so, for each guess we could average the amount of information (reduction of entropy) that each grey-green-yellow pattern that could arise from that guess would yield. Then choose the guess that has the highest average gain of information. To compute the amount of information that a particular pattern would yield with a given guess we could enumerate all the words that would satisfy that pattern that have not been guessed yet and compute the entropy of the set.