CS3236 Term Project:
Gambling and Data Compression

Joyce Yeo Shuhui

AY19/20 Sem2

1 Introduction

In Data Compression, the goal is to encode sequences such that the encoded lengths can be minimised. However, in techniques such as Arithmetic Coding, it is necessary to have a good estimate of the true distribution of the symbols in the source. This notion of having a good estimate of some distribution also features prominently in gambling. More specifically, in horse racing the gambler has to have a good estimate of the distribution of winning probabilities of the horses.

We will first introduce the terminology used in understanding Horse Racing from an information-theoretic perspective (Section 2). Then, we will derive some information-theoretic results on optimising the gambling process (Section 3). Lastly, we will show how the new perspective on gambling is related to data compression (Section 4).

2 Terminology in Horse Racing

Suppose that the gambler has 1 dollar to bet in a race with m horses. The gambler places the bet before the race, on horse i. For every horse i, there will be a known payout, or **odds**, o_i. Suppose that horse i (the horse the gambler bet on) wins, then the gambler is paid o_i times his bet amount, effectively multiplying his wealth by a factor of o_i. The **winning probability** of horse i, p_i, is the probability of horse i winning the race. Hence, the gambler gains o_i with a probability of p_i, so the expected gain is $p_i o_i$.

The gambler may also split his bet across multiple horses, to minimise the chances of walking away with nothing. The **betting proportion** on horse i, b_i, is the fraction of money invested into horse i. We want the betting proportions to satisfy the following properties:

- For all betting proportions, $b_i \geq 0$. It is possible that a horse gets no bets.
- $\Sigma b_i = 1$. The gambler spends all his money on betting in every race.

We have introduced three probability distributions involved in the horse race:

- P, the distribution of the horses' winning probabilities.
- O, the distributions of the horses' winning odds. This is determined by the system taking in bets (e.g. bookie) before the race starts.
- B, the distribution of how the gambler allocates bets on different horses.

Only B is controlled by the gambler.

2.1 Wealth Relative - multiplication factor for each race

When horse i wins, the gambler is paid o_i times the bet amount on horse i. Given a total bet of one dollar, the bet amount on horse i is b_i and the total winnings is $b_i o_i$. Since the gambler started with one dollar, his total wealth was multiplied by $b_i o_i$. This multiplication factor is called the **wealth relative**.

Let X_j denote the outcome of horse race j. $X_j = i$ if and only if horse i won in race j. The wealth relative of a race outcome, $S(X_j)$ is the factor the gambler's wealth is multiplied by in race j. We have that:

$$S(X_j) = b_{X_j} o_{X_j} \tag{1}$$

Example: If horse i won in race j, then $S(X_j) = b_i o_i$.

Combining the gains over multiple races

Often, the gambler will not stop at one race but play multiple rounds. We make two assumptions about the race conditions:

- The distributions of P, O and B remain the same across all races. The winning chances and payouts of the horses do not change. The gambler always bets a fixed proportion of the money b_i on horse i.
- The gambler always bets all the remaining money in every race. No money is kept between any races.

Then, with n independent races, the total wealth relative accumulated is the product of all the wealth relatives.

$$S_n = \Pi_{j=1}^{n} S(X_j) \tag{2}$$

After the first race, the gambler has $S(X_1)$ times his original bet. In the second race, he bets all his winnings and leaves with $S(X_1)S(X_2)$ times his original bet. The gambler's strategy should maximise the wealth relative of each race, consequently maximising the overall gains across n races.

2.2 Doubling Rate versus Wealth Relative

Recall that our goal is to draw comparisons to data compression. In data compression, we measure entropy in bits and use the logarithm of the amount of information. Hence, instead of considering the wealth relative, we apply a logarithmic transformation to the wealth relative. The resulting metric is the **doubling rate** of the gambler's wealth. A race with a doubling rate of 1 has a wealth relative of 2. A unit increase in the doubling rate represents a doubling in the wealth relative.

In horse races, the distribution of odds O over the horses is fixed for a race. Hence, the doubling rate is a function of the gambler's betting strategy B and horse winning probability of the horses P. We use the notation $W(B, P)$ for the doubling rate of a race. After each race, the gambler's wealth is multiplied by a factor of $2^{W(B,P)}$. With n independent races, the total wealth is multiplied by a factor of $(2^{W(B,P)})^n = 2^{nW(B,P)}$.

Then, we have that:

$$\begin{aligned} W(B,P) &= \frac{1}{n} log S_n \\ &= \frac{1}{n} \Sigma_{j=1}^{n} [log S(X_j)] \\ &= E(log S(X)) - (a) \\ &= \Sigma_{i=1}^{m} [p_i * log(b_i o_i)] - (b) \end{aligned} \tag{3}$$

(a) - When there are sufficiently many races n, then the mean over multiple i.i.d. $S(X)$ is the same as the expected value of $S(X)$. (b) uses Equation 1.

3 Gambler's Strategy: Maximising Doubling Rate

The maximum gains is achieved when $W(B, P)$ is maximised. Recall that only the distribution of B is within the gambler's control. Hence, the **optimal doubling rate**, $W^*(P)$, is the maximum value of $W(B, P)$ over all betting profiles B:

$$W^*(P) = max_B[W(B, P)] \tag{4}$$

We consider and interpret the arithmetic equivalent of $W(B, P)$ to find $W^*(P)$.

$$\begin{aligned} W(B, P) &= \Sigma_{i=1}^{m}[p_i log(b_i o_i)] \\ &= \Sigma_i[p_i log(\frac{b_i}{p_i} p_i o_i)] \\ &= \Sigma_i[p_i log(o_i)] + \Sigma_i[p_i log(p_i)] + \Sigma_i[log(\frac{b_i}{p_i})] \\ &= \Sigma_i[p_i log(o_i)] - H(P) - D(P||B) \end{aligned} \tag{5}$$

Since the distributions of P and O are not in the gambler's control, maximising $W(B, P)$ only depends on minimising the divergence $D(P||B)$.

Recall that $D(P||B) \geq 0$, with equality if and only if the distributions of P and B are identical. Hence, the maximum doubling rate is when the gambler makes the bet $b_i = p_i$ for all horses i and $D(P||B) = 0$. This form of betting is known as **proportional gambling**, and gives the log-optimal doubling rate of:

$$W^*(P) = \Sigma_i[p_i log o_i] - H(P) \tag{6}$$

3.1 Fair Odds: Betting system does not take any cuts

These results on there being guaranteed winnings is contrary to common knowledge that gambling results in long-term losses. In real-life gambling, the odds are unfair and betting system will take a cut. The expected wealth relative is less than 1, and the expected doubling rate is negative regardless of the distributions of P, how the horses win, and B, how the gambler bets.

Arithmetically, there are **fair odds** when $\Sigma_{i=1}^{m}\frac{1}{o_i} = 1$. We can find the **bookie's estimate of the winning probabilities** of the horses, $r_i = \frac{1}{o_i}$. When $\Sigma_{i=1}^{m} r_i = 1$, the bookie's distribution is fair and follows the rules of a normal probability distribution.

Then, we have that:

$$\begin{aligned} W(B,P) &= \Sigma_{i=1}^{m}[p_i log(b_i o_i)] \\ &= \Sigma_i[p_i log(\frac{b_i}{p_i}\frac{p_i}{r_i})] \\ &= D(P||R) - D(P||B) \end{aligned} \tag{7}$$

The gambler can only make only when $D(P||R) > D(P||B)$. This is when the bookie's estimate R is farther from the true distribution of P than the gambler's estimate B is.

Unfair Scenario in Reality

The betting is sub-fair (unfair) when $\Sigma_i r_i < 1$. In the unfair case, the bookie assigns a non-zero probability of $1 - \Sigma_i r_i$ where no horse wins. Since the gambler cannot bet on this event where no horse wins, the gambler is disadvantaged.

3.1.1 Uniform fair odds: All horses have the same payout

Consider the special case when the odds are **fair and uniform**. Then, $o_i = m$ for all horses i. Using Equation 6, we get:

$$\begin{aligned} W^*(P) &= log(m) - H(P) \\ H(P) + W^*(P) &= log(m) \end{aligned} \tag{8}$$

For any distribution of winning probabilities P, the sum of the entropy and optimal doubling rate is constant. Increasing the entropy of P by one bit will half the maximum amount the gambler can win.

3.2 Side Information in Gambling

In reality, gambling with side information can improve the gambler's gains. The information-theoretic equivalent is that side information improves the gambler's estimate of P, reducing $D(P||B)$ and increasing the doubling rate $W(B,P)$. Suppose that there is side information Y. Since the gambler makes bets based on side information, we longer consider b_i but $b_{i|y}$. This represents the proportion of money bet on horse i given that side information y was observed. Furthermore, we consider the joint probability between winning probabilities and side information $p_{i,y}$ instead of p_i. Then, we have:

$$\begin{aligned} W(P,B|Y) &= \Sigma_{i,y}[p_{i,y} * log(b_{i|y} o_i)] \\ &= \Sigma_{i,y}[p_{i,y} * log(o_i)] - H(P|Y) - D(P|Y||B) - (a) \\ &= \Sigma_i[p_i * log(o_i)] - H(P|Y) - D(P|Y||B) \end{aligned} \tag{9}$$

(a) uses the same techniques as in Equation 5. Then, instead of considering the optimal doubling rate $W^*(P)$, we consider the optimal doubling rate given side information Y, $W^*(P|Y)$.

$$\begin{aligned} W^*(P|Y) &= max_B[W(P, B|Y)] \\ &= \Sigma_i[p_i * log(o_i)] - H(P|Y) - (a) \end{aligned} \tag{10}$$

(a) uses the same reasoning as in Equation 6. The doubling rate with side information is maximised when the relative entropy between $P|Y$ and B is 0. In this case, the gambler also bets conditionally proportionally given side information y, where $b_{i|y} = p_{i|y}$.

Furthermore, the side information improves the gambler's doubling rate by:

$$\begin{aligned} \Delta W &= W^*(P|Y) - W^*(P) \\ &= H(P) - H(P|Y) \\ &= I(P; Y) \end{aligned} \tag{11}$$

The higher the mutual information between the true winning probability P and the side information Y, the higher the optimal doubling rate. The most common example of side information is the race history. Knowing which horses won previous races gives the gambler a better estimate of which horses will win future races. However, if the side information is independent of the winning probabilities of the horses, then there is no change to the optimal doubling rate.

4 Applying Gambling to Data Compression

Using the information-theoretic understanding of gambling, we will analyse the data compression problem and draw connections between gambling and data compression.

4.1 The English Language

In class, we have seen how the English language does not have a high entropy. The contextual information from preceding characters and preceding words is useful is predicting what the next letter(s) would be. Then, even when a message is encoded into a short sequence of characters, it is still possible to deduce the full message. For instance, "fll n th blnks" can be decoded to "fill in the blanks" easily. Providing more context is analogous to providing more side information, and will improve the probability of decoding the message correctly.

4.2 Using Gamblers in Data Compression

We rethink the data compression problem in terms of a betting sequence. Suppose that an n-character sequence needs to be encoded. Let the characters by X_1, X_2, ..., X_n drawn from some source. The characters X_i all belong to the same English alphabet, which we take to be of size 26 letters + space = 27. Then, we can treat the n characters as n races. Each race has m horses, $m = 27$.

However, to simplify our calculations and without loss of generality, we will work with a binary sequence of length n instead. Given a sequence of binary values, the gambler makes sequential bets based on the previous observed outcomes. The previous letters become the contextual side information for the gambler, analogous to the horse racing history. For example, the gambler first makes a bet on X_1 using b_{X_1}. Then, the gambler makes a bet on X_2 using the observed value of x_1, $b_{X_2|x_1}$. This gives us conditional bets made with the following properties:

- $b_{X_j|x_1,x_2,...,x_{j-1}} \geq 0$, where the gambler makes a bet given all the previous outcomes.
- $\Sigma_{x_j}[b_{X_j|x_1,x_2,...,x_{j-1}}] = 1$, or equivalently $\Sigma_{x\in\{0,1\}}[b_{x|x_1,x_2,...,x_{j-1}}] = 1$.

Given an n-character sequence $x_1, x_2, ..., x_n$, we have that:

$$\Pi_{j=1}^{n}[b_{x_j|x_1,x_2,...,x_{j-1}}] = b_{x_1,x_2,...,x_n} \tag{12}$$

The last result, $b_{x_1,x_2,...,x_n} = b_x(n)$ can be interpreted as the probability of betting on a particular n-character sequence. There are 2^n possible such sequences to be bet on. For each of these sequences $x(n) = (x_1, x_2, ..., x_n)$, there is a probably $b_{x(n)}$ associated with betting on that sequence.

For each sequence $x(n)$, we can also calculate the expected wealth from betting on that sequence. We let the odds be uniform, so $o_i = 2$.

$$\begin{aligned} S(x(n)) &= \Pi_{j=1}^{n} b_{X_j|x_1,x_2,...,x_{j-1}} * o_{X_j} \\ &= 2^n * b_{x(n)} \end{aligned} \tag{13}$$

4.3 Encoding Algorithm

The encoding algorithm for an n-character sequence works as follows. Suppose that $x(n)$ is to be encoded:

1. Calculate the probability of betting on that sequence, $b_{x(n)}$.
2. Arrange the sequences in lexicographical order, so 01001 comes before 10100.
3. Calculate the cumulative probability of betting on $x(n)$ and all lexicographically smaller sequences. This new value is given by:

$$F(x(n)) = \Sigma_{x'(n)\leq x(n)} b_{x'(n)} \tag{14}$$

4. Then, calculate the expected wealth from betting on $x(n)$, $S(x(n))$.
5. Use the expected wealth to find the **compression size** given by:

$$k = \lceil n - log[S(x(n))] \rceil \tag{15}$$

6. Express F(x(n)) as a binary decimal to k-place accuracy, rounded down:

$$\lfloor F(x(n)) \rfloor = .c_1c_2...c_k \tag{16}$$

7. The sequence $c(k) = (c_1, c_2, ..., c_k)$ is the codeword for the original sequence $x(n)$.

4.4 Decoding Algorithm

Given the codeword $c(k)$, the decoder has to retrieve the original n-character sequence:

1. The decoder uses a gambler with the same betting behaviour as the encoding gambler. That is, $b_{X_j|x_1,x_2,...,x_{j-1}}$ is fixed for all $x_j \in X_j$ across all j from 1 to n.
2. Lexicographically sum the values of $b_{x(n)}$ until the cumulative sum first exceeds $c(k)$. Example: Given $n = 4$, sum b_{0000}, then b_{0001} and b_{0010}, until the value first exceeds $c(k)$.
3. The sequence $x(n)$ whose cumulative sum first exceeds $c(k)$ is the decoded message.

The value of $x(n)$ will be correct because the betting behaviours in the encoding gambler and decoding gambler were the same. Hence, both encoder and decoder have the same $b_{x(n)}$ value for all $x(n)$. The cumulative sum can then be unambiguously calculated.

4.5 Evaluation of Gambling-Data Compression Scheme

With the gambling-data compression scheme, it is possible to compress a message of length n to a message of length k. From Equation 15, the number of bits used is $k = \lceil n - log(S_n(x(n))) \rceil$ so the **number of bits saved** is:

$$n - k = \lfloor log[S(x(n))] \rfloor \tag{17}$$

Hence, maximising S(x(n)) will maximise the number of bits saved in data compression.

<u>Optimal Scenario</u>

In the optimal case, proportional gambling is applied. Then, for every sequence $x(n)$, $b_{x(n)} = p_{x(n)}$, where $p_{x(n)}$ is the joint probability of that n-character sequence among all 2^n possible sequences. From Equation 13, we get the optimal wealth relative on an n-character sequence as:

$$S^*(x(n)) = 2^n * p_{x(n)} \tag{18}$$

Then, the expected number of bits required for the compressed data, k, is given by:

$$\begin{aligned} E(k) &= E[\lceil n - log[S(x(n))] \rceil] \\ &= n + E[\Sigma_{x(n)}(p_{x(n)} * \lceil -log[2^n * p_{x(n)}] \rceil)] \\ &= E[\Sigma_{x(n)}(p_{x(n)} * \lceil log\frac{1}{p_{x(n)}} \rceil)] \\ &= \lceil H(X_1, X_2, ..., X_n) \rceil \\ &\leq H(X_1, X_2, ..., X_n) + 1 \end{aligned} \tag{19}$$

Applying the gambling technique to a message of size n can give an optimal compression size of at most $H(X_1, X_2, ..., X_n) + 1$, which is close to the Shannon limit of $H(X_1, X_2, ..., X_n)$.

<u>Practical Use</u>

Applying this scheme in practice would require the computation of the cumulative wealth. This is either time-intensive or storage-intensive, if we choose to pre-compute the values of $F(x(n))$. Furthermore, to achieve the optimal compression size, a good estimate of the true distribution of the English language is necessary. Both of these assumptions may be unrealistic.

4.6 Estimating the Entropy of the English Language

We can extend the gambling-data compression connection to get other useful results such as estimating the entropy of the English language, $H(\chi)$.

Given a sequence of characters, the gambler has to make bets on the next possible letter. By considering this problem, we are able to estimate the value of $H(\chi)$. Experimentally, subjects will be given sequences of English characters and have to predict the next character. The payout for all characters is uniform, where $o_i = 27$. Hence, the wealth relative is the portion bet on the correct character i multiplied by the payout or 27. This statement is equivalent to $S(X) = b_X * o_X$ in Equation 1.

We modify Equation 13 to consider the 27-character problem. The wealth gain is given by: $S_n = 27^n * b_{X_1,X_2,...,X_n}$. Then, we consider the expected doubling rate using Equation 5:

$$\begin{aligned} E[\frac{1}{n}log(S_n)] &= log(27) - \frac{1}{n}H(X_1, X_2, ..., X_n) - \frac{1}{n}D(P_{X^n}||B_{X^n}) - (a) \\ &\leq log(27) - \frac{1}{n}H(X_1, X_2, ..., X_n) \\ &\leq log(27) - H(\chi) \end{aligned} \tag{20}$$

(a) uses the notion of average entropy and average relative entropy over n characters.

Hence, the entropy of English is upper-bounded by $log(27) - E[\frac{1}{n}log(S_n)]$. By using gambling experiments to obtain a value for S_n, the entropy of English can be upper-bounded.

5 Conclusion

This project explored the notion of gambling in horse races from an information-theoretic perspective. Unfortunately, some assumptions such as the odds being fair are unrealistic.

Then, we applied the results from gambling to data compression. The connection between both problems is a reliance on the true distribution of the source symbols (in data compression) and winning probabilities of the horses (in gambling). While the encoding scheme proposed has practical limitations, it achieves a very good theoretical bound close to the Shannon limit.

6 References

1. Book section 6, *Elements of Information Theory* 2nd Edition, Thomas Cover