



Entropy and Decisions

CSC401/2511 – Natural Language Computing – Spring 2026
University of Toronto

Entropy

LMs and Information Theory

- LMs may be evaluated **extrinsically** through their embedded performance on other tasks
- An LM may be evaluated **intrinsically** according to how accurately it predicts language
- **Information Theory** was developed in the 1940s for **data compression and transmission**
- Many of the concepts, chiefly **entropy**, apply directly to LMs

Information

- Imagine Darth Vader is about to say either “yes” or “no” with **equal** probability.
 - You don’t know what he’ll say.
- You have a certain amount of **uncertainty** – a lack of information.



Darth Vader is © Disney
And the prequels and Rey/Finn Star Wars suck

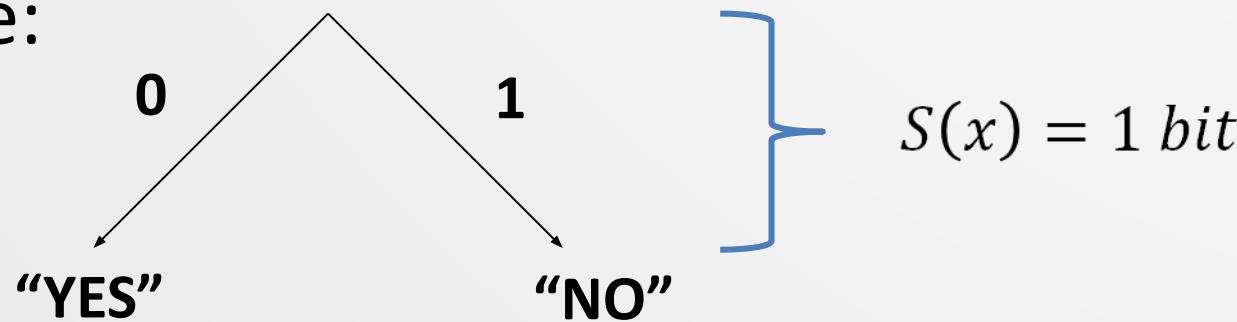
Information

- Imagine you then **observe** Darth Vader saying “no”
- You’d be **surprised**: he could’ve said “yes”
- Your uncertainty is **gone**; you’ve **received** **information**.
- **How much** information do you **receive** about event x when you observe it?



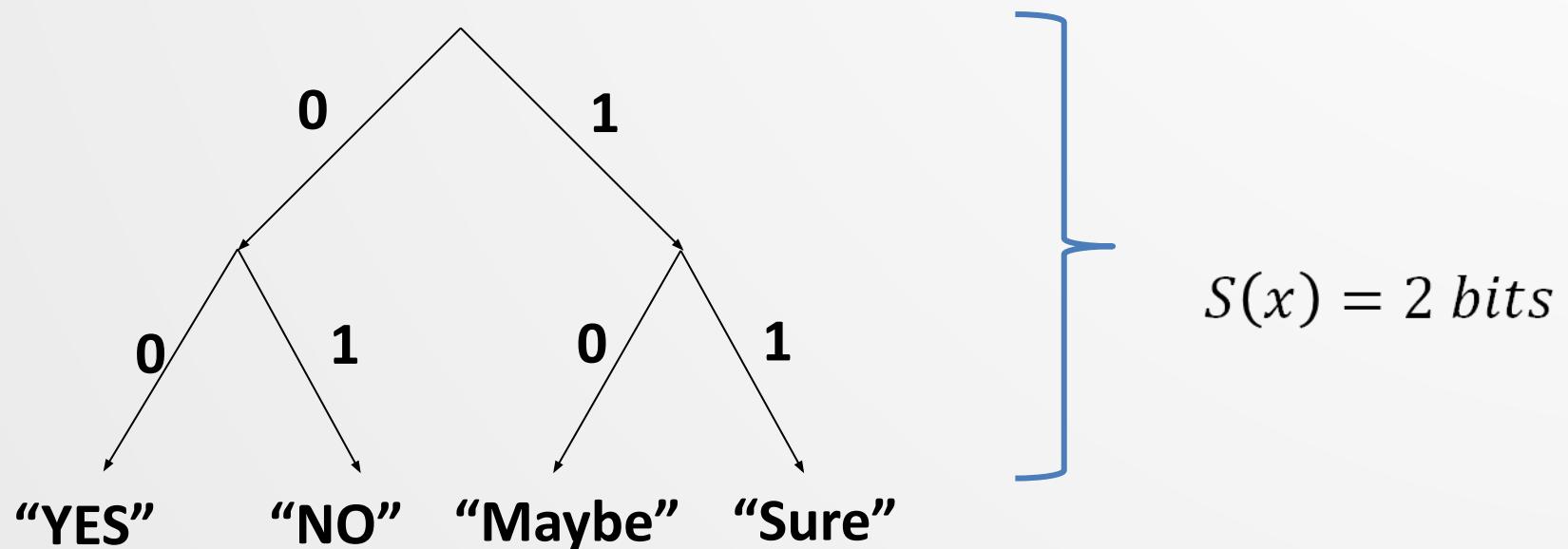
Information

- Imagine communicating the outcome in binary
- The amount of information is the size of the message
- What's the **minimum, average** number of bits needed to encode any outcome?
- **Answer: 1**
- Example:



Information

- What about 4 equiprobable words?



- In general $S(x) = \log_2 \left(\frac{1}{P(x)} \right) = -\log_2 P(x)$

Information

- Imagine Darth Vader is about to roll a **fair** die.
- You have **more uncertainty** about an event because there are **more** (equally probable) **possibilities**.
- You **receive** more information when you observe it.
- You are more **surprised** by any given outcome.



$$\begin{aligned} S(x) &= \log_2 \frac{1}{P(x)} \\ &= \log_2 \frac{1}{1/6} \approx \underline{\underline{2.58 \text{ bits}}} \end{aligned}$$

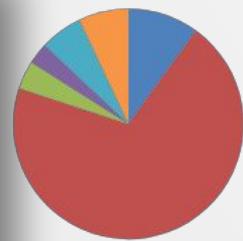
Information can be additive

- One property of $S(x) = \log_2 \frac{1}{P(x)}$ is additivity.
- From k **independent** events $x_1 \dots x_k$:
 - Does $S(x_1 \dots x_k) = S(x_1) + S(x_2) + \dots + S(x_k)$?
- The answer is yes!

$$\begin{aligned} S(x_1 \dots x_k) &= \log_2 \frac{1}{P(x_1 \dots x_k)} \\ &= \log_2 \frac{1}{P(x_1) \dots P(x_k)} = \log_2 \frac{1}{P(x_1)} + \dots + \log_2 \frac{1}{P(x_k)} \\ &= S(x_1) + S(x_2) + \dots + S(x_k) \end{aligned}$$

Events with unequal information

- Events are not always equally likely
- Surprisal will therefore be dependent on the event
- How surprising is the distribution overall?
 - Suppose you **still** have 6 outcomes that are possible – **but** you’re fairly sure it will be ‘No’.
 - We expect to be less surprised on **average**



- Yes (0.1)
- No (0.7)
- Maybe (0.04)
- Sure (0.03)
- Darkside (0.06)

Entropy

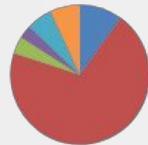
- **Entropy**: *n.* the average uncertainty/information/surprisal of a (discrete) random variable X .

$$H(X) = \sum_x P(x) \log_2 \frac{1}{P(x)}$$

Expectation over X

- A lower bound on the average number of bits necessary to encode X (more on this later)

Entropy – examples



- Yes (0.1)
- No (0.7)
- Maybe (0.04)

$$\begin{aligned} H(X) &= \sum_i p_i \log_2 \frac{1}{p_i} \\ &= 0.7 \log_2(1/0.7) + 0.1 \log_2(1/0.1) + \dots \\ &= 1.542 \text{ bits} \end{aligned}$$

There is **less** average uncertainty when the probabilities are ‘skewed’.

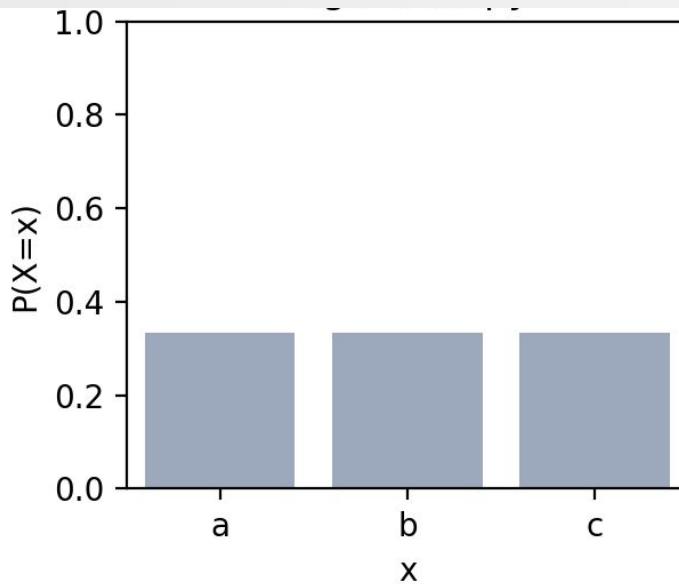


- 1 ■ 2 ■ 3 ■ 4 ■ 5 ■ 6

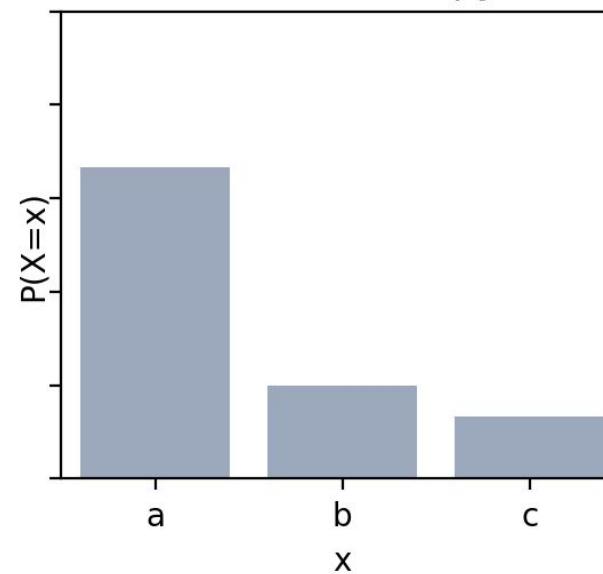
$$\begin{aligned} H(X) &= \sum_i p_i \log_2 \frac{1}{p_i} = 6 \left(\frac{1}{6} \log_2 \frac{1}{1/6} \right) \\ &= 2.585 \text{ bits} \end{aligned}$$

Entropy characterizes the distribution

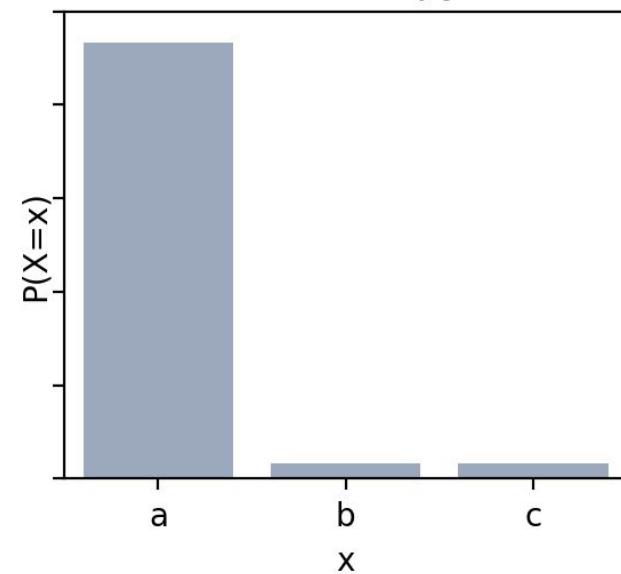
- **Flatter** distributions \Rightarrow **higher** entropy \Rightarrow **hard** to predict
- **Peaky** distributions \Rightarrow **lower** entropy \Rightarrow **easy** to predict



High



Medium



Low

Bounds on entropy

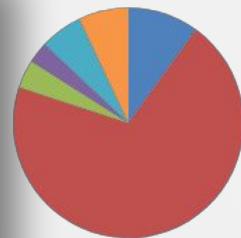
- **Maximum**: uniformly distributed X_1 . Given V choices,

$$H(X_1) = \sum_i p_i \log_2 \frac{1}{p_i} = \sum_i \frac{1}{V} \log_2 \frac{1}{1/V} = \log_2 V$$

- **Minimum**: only one choice, $H(X_2) = p_i \log_2 \frac{1}{p_i} = 1 \log_2 1 = 0$

Coding with fewer bits is better

- If we want to **transmit** Vader's words **efficiently**, we can **encode** them so that **more probable words** require **fewer bits**.
 - On **average**, fewer bits will need to be transmitted.



- Yes (0.1)
- No (0.7)
- Maybe (0.04)
- Sure (0.03)
- Darkside (0.06)

Word (sorted)	Linear Code	Probability	Huffman Code
No	000	0.7	0
Yes	001	0.1	100
Destiny	010	0.07	101
Darkside	011	0.06	110
Maybe	100	0.04	1111
Sure	101	0.03	1110

$$\text{Average codelength (Huffman)} = 1*0.7+3*(0.1+.07+.06)+4*(.04+.03) = \text{1.67 bits} > 1.54 \text{ bits} \approx H(X)$$

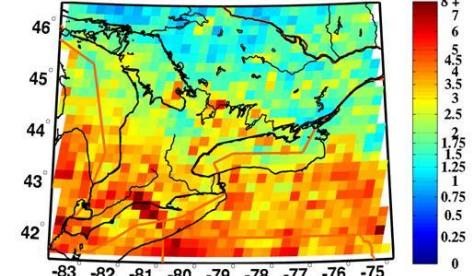
The entropy rate of language

- Can we use entropy to measure how predictable language is?
- Imagine that language follows an LM P which infinitely generates one word after another: $X = X_1, X_2, \dots$
 - A corpus c is a prefix of x
- Uh oh: as $N \rightarrow \infty, H(X) = \infty$
- Instead, we take the per-word **entropy rate**

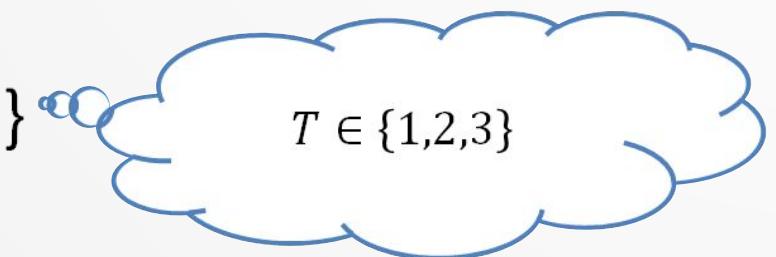
$$H_{rate}(X) = \lim_{N \rightarrow \infty} \frac{1}{N} H(X_1, \dots, X_N) \leq \log_2 V$$

- How do we handle more than one variable?
- How do we evaluate $P(x)$?

Entropy of several variables



- Consider the vocabulary of a meteorologist describing Temperature and Wetness.
 - Temperature $\in \{\text{hot, mild, cold}\}$
 - Wetness $\in \{\text{dry, wet}\}$



$$P(W = \text{dry}) = 0.6, \\ P(W = \text{wet}) = 0.4$$

$$H(W) = 0.6 \log_2 \frac{1}{0.6} + 0.4 \log_2 \frac{1}{0.4} = \mathbf{0.970951 \text{ bits}}$$

$$P(T = \text{hot}) = 0.3, \\ P(T = \text{mild}) = 0.5, \\ P(T = \text{cold}) = 0.2$$

$$H(T) = 0.3 \log_2 \frac{1}{0.3} + 0.5 \log_2 \frac{1}{0.5} + 0.2 \log_2 \frac{1}{0.2} = \mathbf{1.48548 \text{ bits}}$$

But W and T are *not* independent,
 $P(W, T) \neq P(W)P(T)$

Joint entropy

- **Joint Entropy**: *n.* the **average** amount of information needed to specify **multiple** variables **simultaneously**.

$$H(X, Y) = \sum_x \sum_y p(x, y) \log_2 \frac{1}{p(x, y)}$$

- **Hint:** this is *very* similar to univariate entropy – we just replace univariate probabilities with joint probabilities and sum over everything.

Entropy of several variables

- Consider joint probability, $P(W, T)$

	cold	mild	hot	
dry	0.1	0.4	0.1	0.6
wet	0.2	0.1	0.1	0.4
	0.3	0.5	0.2	1.0

- Joint entropy**, $H(W, T)$, computed **as a sum over the space of joint events** ($W = w, T = t$)

$$H(W, T) = 0.1 \log_2 \frac{1}{0.1} + 0.4 \log_2 \frac{1}{0.4} + 0.1 \log_2 \frac{1}{0.1} \\ + 0.2 \log_2 \frac{1}{0.2} + 0.1 \log_2 \frac{1}{0.1} + 0.1 \log_2 \frac{1}{0.1} = \mathbf{2.32193 \text{ bits}}$$

Notice $H(W, T) \approx 2.32 < 2.46 \approx H(W) + H(T)$

Entropy given knowledge

- In our example, **joint entropy** of two variables together is **lower** than the **sum** of their **individual** entropies
 - $H(W, T) \approx 2.32 < 2.46 \approx H(W) + H(T)$
- **Why?**
- Information is **shared** among variables
 - There are **dependencies**, e.g., between temperature and wetness.
 - E.g., if we knew **exactly** how **wet** it is, is there **less confusion** about what the **temperature** is ... ?

Conditional entropy

- **Conditional entropy**: *n.* the **average** amount of information needed to specify one variable given that you know another.

$$H(Y|X) = \sum_{x \in X} p(x)H(Y|X = x)$$

- **Comment:** this is the expectation of $H(Y|X)$, w.r.t. x .

Entropy given knowledge

- Consider **conditional** probability, $P(T|W)$

	cold	mild	hot	
dry	0.1	0.4	0.1	0.6
wet	0.2	0.1	0.1	0.4
	0.3	0.5	0.2	1.0

$$P(T|W) = P(W, T)/P(W)$$



	cold	mild	hot	
dry	0.1/0.6	0.4/0.6	0.1/0.6	1.0
wet	0.2/0.4	0.1/0.4	0.1/0.4	1.0

Entropy given knowledge

- Consider **conditional** probability, $P(T|W)$

	cold	mild	hot	
dry	1/6	2/3	1/6	1.0
wet	1/2	1/4	1/4	1.0

- $H(T|W = \text{dry}) = H\left(\left\{\frac{1}{6}, \frac{2}{3}, \frac{1}{6}\right\}\right) = 1.25163$ bits
- $H(T|W = \text{wet}) = H\left(\left\{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\right\}\right) = 1.5$ bits
- Conditional entropy combines these:
$$\begin{aligned} H(T|W) &= [p(W = \text{dry})H(T|W = \text{dry})] + [p(W = \text{wet})H(T|W = \text{wet})] \\ &= 1.350978 \text{ bits} \end{aligned}$$

Equivocation removes uncertainty

- Remember $H(\mathbf{T}) = 1.48548$ bits
- $H(\mathbf{W}, \mathbf{T}) = 2.32193$ bits
- $H(\mathbf{T}|\mathbf{W}) = 1.350978$ bits

• How much does \mathbf{W} tell us about \mathbf{T} ?

- $H(\mathbf{T}) - H(\mathbf{T}|\mathbf{W}) = 1.48548 - 1.350978 \approx 0.1345$ bits
- Well, a little bit!

 **Entropy** (i.e., confusion) about temperature is **reduced** if we **know** how wet it is outside.

• Perhaps T is more informative?

- Consider **another** conditional probability, $P(W|T)$

		mild	hot
	0.1/ 0.3	0.4/ 0.5	0.1/ 0.2
wet	0.2/ 0.3	0.1/ 0.5	0.1/ 0.2
	1.0	1.0	1.0

- $H(W|T = \text{cold}) = H\left(\left\{\frac{1}{3}, \frac{2}{3}\right\}\right) = 0.918295$ bits
- $H(W|T = \text{mild}) = H\left(\left\{\frac{4}{5}, \frac{1}{5}\right\}\right) = 0.721928$ bits
- $H(W|T = \text{hot}) = H\left(\left\{\frac{1}{2}, \frac{1}{2}\right\}\right) = 1$ bit
- **$H(W|T) = 0.8364528$ bits**

A little bit of knowledge still removes uncertainty, but ...

- $H(T) = 1.48548$ bits
- $H(W) = 0.970951$ bits
- $H(W, T) = 2.32193$ bits
- $H(T|W) = 1.350978$ bits
- $H(T) - H(T|W) \approx 0.1345$ bits

Previously
computed

- How much does T tell us about W on average?
 - $H(W) - H(W|T) = 0.970951 - 0.8364528 \approx 0.1345$ bits
- Interesting ... is that a coincidence?

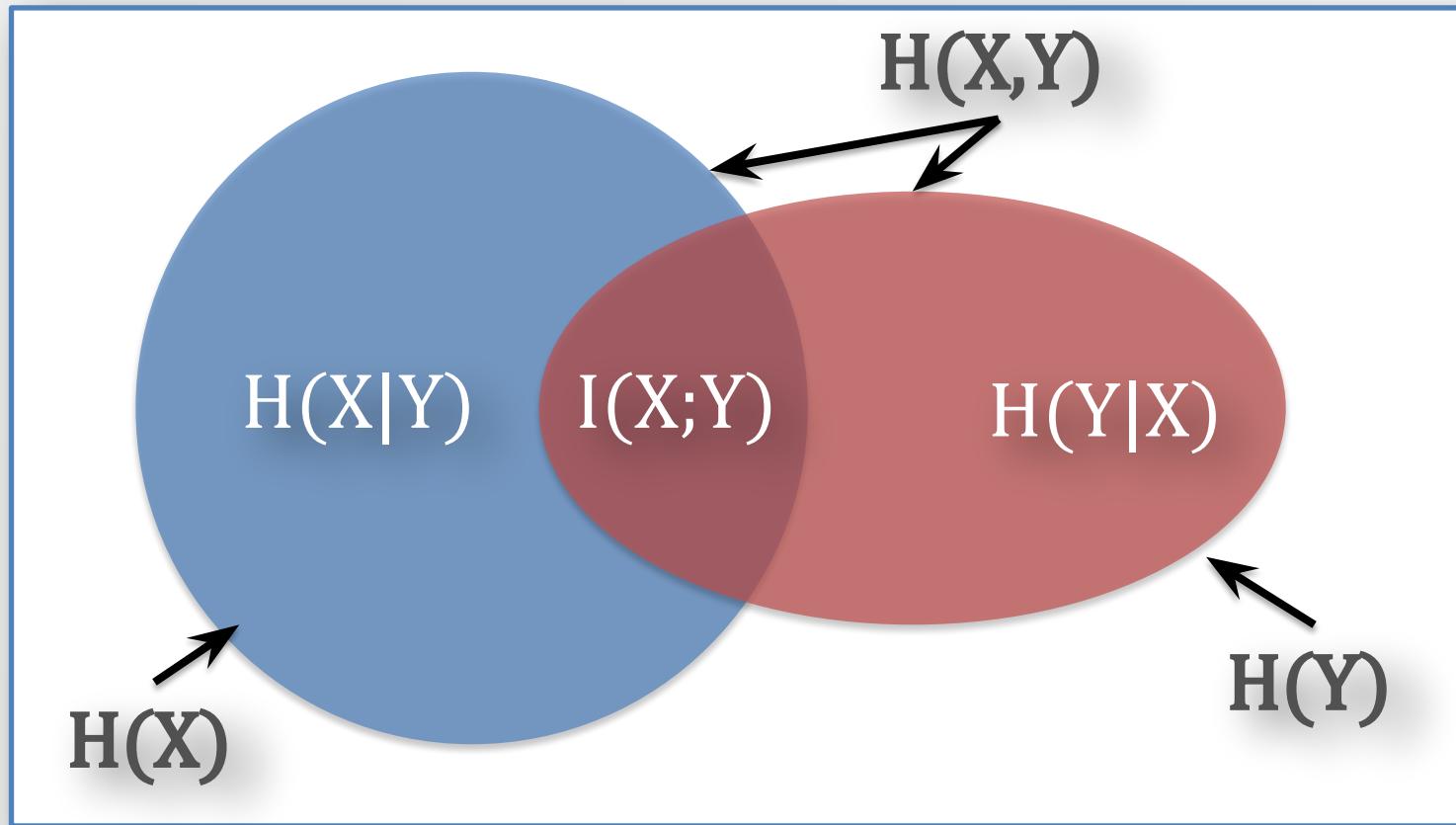
Mutual information

- **Mutual information:** *n.* the **average** amount of information **shared** between variables.

$$\begin{aligned} I(X; Y) &= H(X) - H(X|Y) = H(Y) - H(Y|X) \\ &= \sum_{x,y} p(x, y) \log_2 \frac{p(x, y)}{p(x)p(y)} \end{aligned}$$

- **Hint:** The amount of uncertainty **removed** in variable X if you know Y .
- **Hint2:** If X and Y are **independent**, $p(x, y) = p(x)p(y)$, then $\log_2 \frac{p(x, y)}{p(x)p(y)} = \log_2 1 = 0 \ \forall x, y$ – **there is no mutual information!**

Relations between entropies



$$H(X, Y) = H(X) + H(Y) - I(X; Y)$$

Returning to language

- Recall $H_{rate}(X) = \lim_{N \rightarrow \infty} \frac{1}{N} H(X_1, X_2, \dots, X_N)$
- Now we have

$$H(X_1, X_2, \dots, X_N) = \sum_{x_1, \dots, x_N} P(x_1, \dots, x_N) \log_2 \frac{1}{P(x_1, \dots, x_N)}$$

- But we still don't know how to compute $P(\dots)$
- We will approximate the log terms with our trained LM Q

Cross-entropy

- **Cross-entropy** measures the uncertainty of a distribution Q of samples drawn from P

$$H(X; Q) = \sum_x P(x) \log_2 \frac{1}{Q(x)}$$

- As Q nears P , cross-entropy nears entropy
- We pay for this mismatch with added uncertainty
 - More on this shortly

Estimating cross-entropy

- We can evaluate Q but not P
- **But** corpus $c = x_1, \dots, x_N$ is drawn from P !
- Let s_1, s_2, \dots, s_M be c 's sentences where $\sum_m |s_m| = N$

$$\begin{aligned} H_{rate}(X) &\approx \frac{1}{N} H(X_1, \dots, X_N) && \leftarrow (\text{large } N) \\ &\approx \frac{1}{N} H(X_1, \dots, X_N; Q) && \leftarrow (Q \approx P) \\ &\approx \frac{1}{N} \log_2 \frac{1}{Q(c)} && \leftarrow (\text{it happened!}) \\ &\approx \frac{1}{N} \sum_{m=1}^M \log_2 \frac{1}{Q(s_m)} \end{aligned}$$

= Negative Log Likelihood (NLL)

- **Aside:** With **time invariance**, **ergodicity**, and $Q = P$, NLL approaches $N \times H_{rate}$ as $N \rightarrow \infty$

Quantifying the approximation

- How well does cross-entropy approximate entropy?
 - **Well** if P and Q are close
 - **Poorly** if P and Q are far apart
- If we can quantify the “closeness” of P and Q , we can quantify how good/bad our NLL estimate is

Quick Recap

- Information $S(x) = \log_2 \left(\frac{1}{P(x)} \right) = -\log_2 P(x)$

- Entropy, Entropy Rate

$$H(X) = \sum_x P(x) \log_2 \frac{1}{P(x)}$$

$$H_{rate}(X) = \lim_{N \rightarrow \infty} \frac{1}{N} H(X_1, \dots, X_N) \leq \log_2 V$$

- Joint Entropy, Conditional Entropy

$$H(X, Y) = \sum_x \sum_y p(x, y) \log_2 \frac{1}{p(x, y)}$$

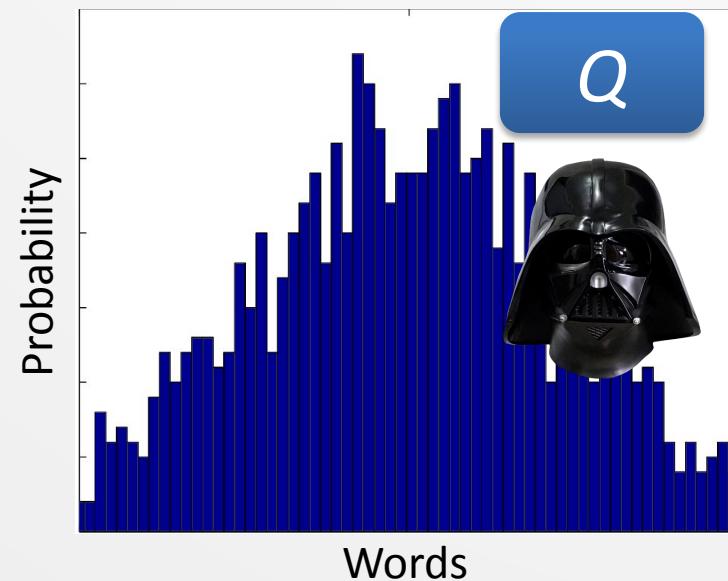
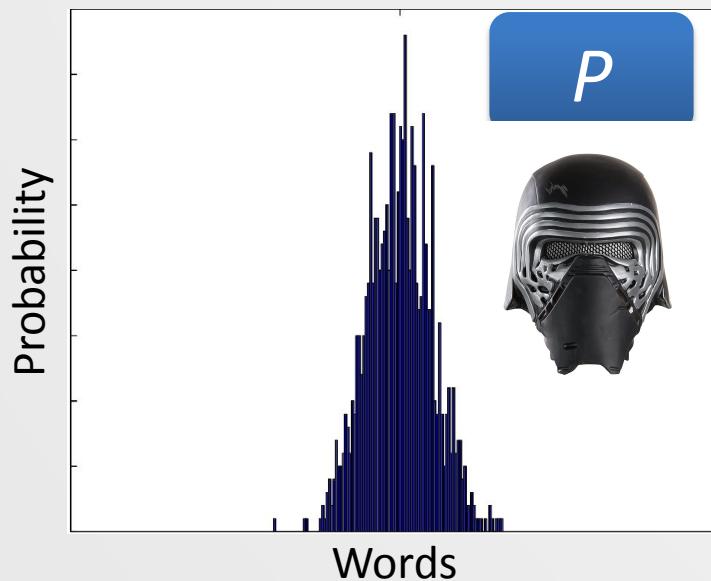
$$H(Y|X) = \sum_{x \in X} p(x) H(Y|X = x)$$

- Mutual Information $I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$
 $= \sum_{x,y} p(x, y) \log_2 \frac{p(x, y)}{p(x)p(y)}$
- Cross-entropy

$$H(X; Q) = \sum_x P(x) \log_2 \frac{1}{Q(x)}$$

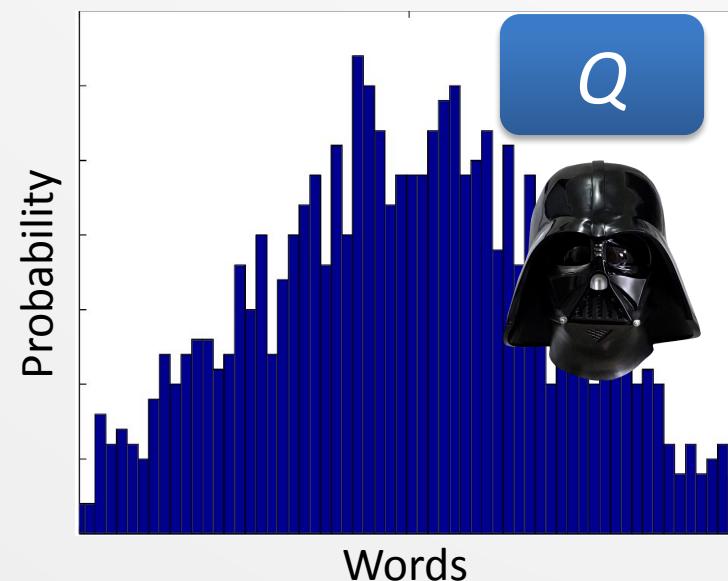
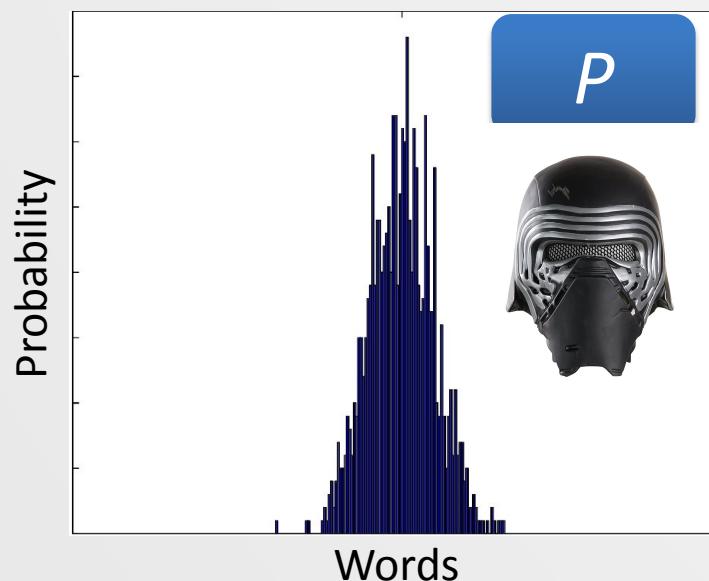
Relatedness of two distributions

- How **similar** are two probability distributions?
 - e.g., Distribution P learned from *Kylo Ren*
Distribution Q learned from *Darth Vader*



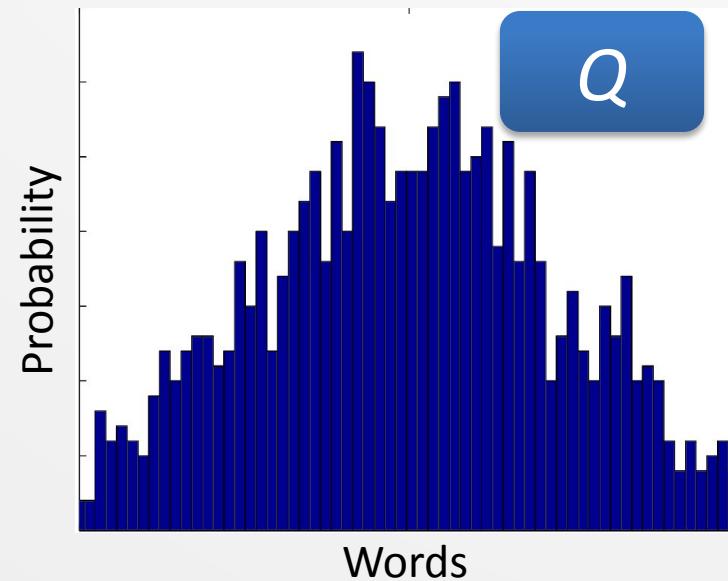
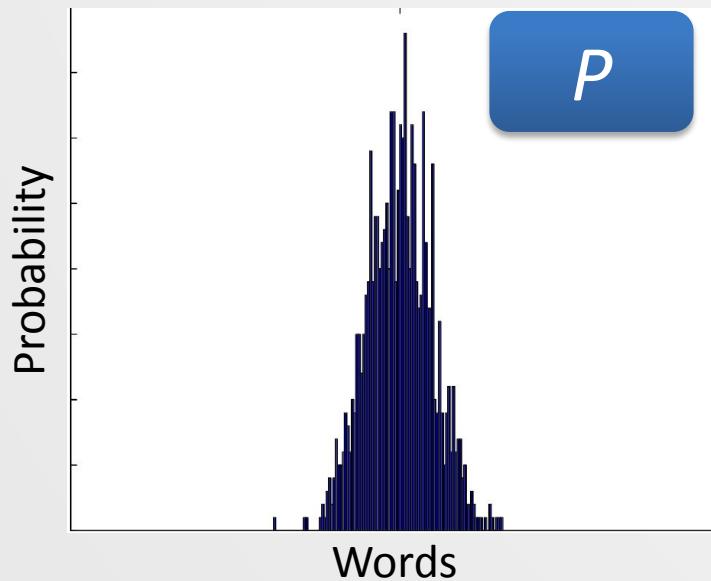
Relatedness of two distributions

- An optimal code based on Vader (Q) instead of Kylo (P) will be less *efficient* at coding symbols that Kylo will say.
- *What is the average number of extra bits required to code symbols from P when using a code based on Q ?*



Kullback-Leibler divergence

- **KL divergence**: *n.* the **average log difference** between the distributions P and Q , relative to Q .
a.k.a. **relative entropy**.
caveat: we assume $0 \log 0 = 0$



Kullback-Leibler divergence

$$D_{KL}(P||Q) = \sum_x P(x) \log_2 \frac{P(x)}{Q(x)}$$

- It is *somewhat* like a ‘**distance**’ :
 - $D_{KL}(P||Q) \geq 0 \quad \forall P, Q$
 - $D_{KL}(P||Q) = 0$ iff P and Q are identical.
- It is **not symmetric**, $D_{KL}(P||Q) \neq D_{KL}(Q||P)$
- **Aside**: normally computed in base e

KL and cross-entropy

- Manipulating KL, we get

$$\begin{aligned} D_{KL}(P||Q) &= \sum_x P(x)\log_2 \frac{1}{Q(x)} - \sum_x P(x)\log_2 \frac{1}{P(x)} \\ &= H(X; Q) - H(X) \geq 0 \end{aligned}$$

- Therefore,

$$\begin{aligned} H_{\text{rate}}(X) &\approx H(X_1, \dots, X_N) \\ &\leq H(X_1, \dots, X_N; Q) \approx NLL(c; Q) \end{aligned}$$

- The NLL is an **approximate upper bound** on $H_{\text{rate}}(X)$

Perplexity

- The intrinsic quality of an LM is often quantified by its perplexity on **held-out** data c by exponentiating its NLL

$$PP(c; Q) = 2^{\frac{1}{N} \sum_{m=1}^M \log_2 \frac{1}{Q(s_m)}} = \left(\prod_{m=1}^M \frac{1}{Q(s_m)} \right)^{1/N}$$

- A uniform Q over a vocabulary of size V gives $PP(c; Q) = V$
 - PP is sort of like an “effective” vocabulary size
- If an LM Q has a lower PP than Q' (for large N), then
 - Q better predicts c
 - $D_{KL}(P||Q) < D_{KL}(P||Q')$
 - $PP(c; Q)$ is a tighter bound on $2^{H_{rate}(X)}$

Perplexity (per token)

- The intrinsic quality of an LM is often quantified by its perplexity on **held-out** data c by exponentiating its NLL

$$PP(c; Q) = 2^{\frac{1}{N} \sum_{m=1}^M \log_2 \frac{1}{Q(s_m)}} = \left(\prod_{m=1}^M \frac{1}{Q(s_m)} \right)^{1/N}$$

- A uniform Q over a vocabulary of size V gives $PP(c; Q) = V$
 - PP is sort of like an “effective” vocabulary size
- If an LM Q has a lower PP than Q' (for large N), then
 - Q better predicts c
 - $D_{KL}(P||Q) < D_{KL}(P||Q')$
 - $PP(c; Q)$ is a tighter bound on $2^{H_{rate}(X)}$

Decisions

Deciding what we know

- (Cross-)entropy, KL divergence, and perplexity can all be used to justify a preference for one method/idea over another
 - “ Q is a better language model than Q' ”
- Engineering statistics are often not enough to be truly meaningful.
 - “My ASR system is 95% accurate on my test data. Yours is only 94.5% accurate! *Heh heh heh*”
 - What if the test data was **biased** somehow?
 - What if our estimates were inaccurate due to simple **randomness**?
 - We need tests to increase our confidence in our results.

Statistical significance testing

Step 1: **State** a hypothesis (and choose a test)

- Decide on the null hypothesis H_0

Step 2: **Compute** some test statistics and associated p-value

- Such as the t -statistic

Step 3: **Reject** H_0 if $p \leq \alpha$, otherwise **do not reject** it

- Significance level α usually ≤ 0.05
- If you can **reject** H_0 , then the result is **significant**

Null hypothesis and *p*-value

- **Null hypothesis** H_0 usually states that “there is no effect”.
 - It is the negation of what you hope for
 - The phrasing of “there is no effect” dictates the appropriate test (and its negation)
 - “The sample is drawn from a normal distribution with some fixed mean”
- You want to **cast doubt** on the plausibility of H_0
 - It’s very unlikely that this measurement would be observed randomly under the H_0
- The ***p*-value** of is the probability that the measured effect occurs under H_0 by chance

Statistical tests

- Here are **some** popular tests (**no need to memorize**)
 - $\bar{X} = \frac{1}{N} \sum_n X_n$ is the sample mean

Test	H_0	Example use case
Two-sided, one-sample t test	$\bar{X} \sim \mathcal{N}(\mu, \sigma)$ for known μ , unknown σ	Whether Elon's average tweet length is different from the average user's ($\mu = 100$)
One-sided, two-sample t test	$\bar{A} \sim \mathcal{N}(\mu_A, \sigma), \bar{B} \sim \mathcal{N}(\mu_B, \sigma)$ for unknown μ_A, μ_B, σ where $\mu_A \leq \mu_B$ (or $\mu_A \geq \mu_B$)	Whether ASR system A (trained N times) makes fewer mistakes than B (trained N times)
One-way ANOVA	$\bar{X}_1, \bar{X}_2, \dots \sim \mathcal{N}(\mu, \sigma)$ for unknown μ, σ	Whether network architecture predicts accuracy
One-sided Mann Whitney U test	$P(A_n > B_{n'}) \leq 0.5$ (or ≥ 0.5)	Whether ASR system A (trained N times) makes fewer mistakes than B (trained N times)



Pitfall 1: parametric assumptions

- **Parametric tests** make assumptions about the **parameters** and **distribution** of RVs
 - Often **normally distributed** with some **fixed variance**
- If **untrue**, H_0 could be **rejected** for spurious reasons
- Must first pass **tests of normality** – difficult with small N
- If non-normal, must use **non-parametric** tests
 - Tend to be less powerful (p -values are higher)

Pitfall 2: multiple comparisons

- Imagine you're flipping a coin to see if it's fair. You claim that if you get 'heads' in 9/10 flips, it's biased.
- Assuming H_0 , the coin is fair, the probability that **one** fair coin would come up heads ≥ 9 out of 10 times is
$$p_1 = 11 \times 0.5^{10} \approx 0.01$$
- **But** the probability that **any of 173** coins hits $\geq \frac{9}{10}$ is
$$p_{173} = 1 - (1 - p_1)^{173} \approx 0.84$$
- The more tests you conduct with a statistical test, the more likely you are to accidentally find spurious (incorrect) significance **accidentally**.

Pitfall 3: effect size

- Just because an effect is reliably measured doesn't make it important
 - Even $\mu_1 = 1$ and $\mu_2 = 1.000000000000001$ can be significantly different
- One must decide whether the purported difference is worth the extra attention
 - There are various measures of **effect size** to support this

More information

- This is a cursory introduction to experimental statistics and hypothesis testing
- You should be aware of their **key concepts** and some of their **pitfalls**
- Before you run your own experiments:
 - Take STA248 “Statistics for computer scientists”
 - Look up stats packages for R, Python
 - Read a book, e.g.:
 - Using multivariate statistics, 7th ed., Tabachnick, Pearson; 2019.
 - Categorical Data Analysis, 3rd ed., Agresti, Wiley, 2013.
 - Ask a statistician for help

Appendix

Everything beyond this slide is **not** on the exam.

Samples, events, and probabilities

- **Samples** are the unique outcomes of an experiment
 - The set of all samples is the **sample space**
 - Examples:
 - What DV could say (“yes” or “no”)
 - The face-up side of a die (1..6)
- **Events** are subsets of the sample space assigned a probability
 - This is usually *any* subset of the sample space
 - Examples:
 - {“yes”}, {“no”}, {“yes”, “no”}, \emptyset
 - The face-up side is even
- The function assigning probabilities to events is the **probability function**

Random variables

- **Random variables (RVs)** are real-valued functions on samples/outcomes of a probability space
- The RV is usually upper-case X while its value is lower x
- Examples:
 - A function returning the sum of face-up sides of N dice
 - A function counting a discrete sample space
 - E.g. “Yes” = 1, “No” = 2
- Like a programming variable, but with uncertainty
 - Let X be defined over samples ω and a, b real
 - $Z = aX + b$ means $\forall \omega: Z(\omega) = aX(\omega) + b$
 - $X = x$ occurs with some probability $P(x)$

PMFs and laziness

- A **probability mass function** (pmf) sums the probabilities of samples mapped to a given RV value

$$P(X = x) = \sum_{\omega \in \Omega_x} P(\{\omega\}), \Omega_x = \{\omega : X(\omega) = x\}$$

- It is often expressed as $P(x)$ or $p(x)$
- If the values of X are 1-to-1 with samples, the pmf is easily confused with the probability function
 - $P(x)$ could be either
 - $P(X = x)$ is the pmf
 - $P(X = \text{yes})$ is an abuse of notation

Expected value

- The **expected value** of an RV is its average (or mean) value over the distribution
- More formally, the expected value of X is the arithmetic mean of its values weighted by the pmf
$$E_X[X] = \sum_x P(X = x) x$$
- $E[\cdot]$ is a **linear operator**
 - $E_{X,Y}[aX + Y + b] = aE_X[X] + E_Y[Y] + b$

Expected value - examples

- What is the average sum of face-up values of 2 fair, 6-sided dice?
- Let X_2 be the sum

2	3	4	5	6	7	8	9	10	11	12
{1,1}	{2,1}	{3,1}	{4,1}	{5,1}	{6,1}	{6,2}	{6,3}	{6,4}	{6,5}	{6,6}
{1,2}	{2,2}	{3,2}	{4,2}	{5,2}	{5,3}	{5,4}	{5,5}	{5,6}		
	{1,3}	{2,3}	{3,3}	{4,3}	{4,4}	{4,5}	{4,6}			
		{1,4}	{2,4}	{3,4}	{3,5}	{3,6}				
			{1,5}	{2,5}	{2,6}					
				{1,6}						

- $E[X_2] = \sum_{x=2}^{12} P(X_2 = x)x = \frac{1}{36}2 + \frac{2}{36}3 + \dots = 7$
- **Alternatively**, let $X_2 = 2X_1$
 - $E[2X_1] = 2E[X_1] = 2 \times 3.5 = 7$