Netflix intro

What data should we encode about each Netflix account holder to help us make effective recommendations?

In machine learning, clustering can be used to group similar data for prediction and recommendation. For example, each Netflix user's viewing history can be represented as a n-tuple indicating their preferences about movies in the database, where n is the number of movies in the database. People with similar tastes in movies can then be clustered to provide recommendations of movies for one another. Mathematically, clustering is based on a notion of distance between pairs of n-tuples.

Data types

Term	**Examples**: (add additional examples from class)
set unordered collection of elements *repetition doesn't matter* *Equal means agree on membership of all elements*	$7 \in \{43, 7, 9\}$ $\quad$ $2 \notin \{43, 7, 9\}$
n-tuple ordered sequence of elements with n "slots" *repetition matters, fixed length* *Equal means corresponding components equal*	
string ordered finite sequence of elements each from specified set *repetition matters, arbitrary finite length* *Equal means same length and corresponding characters equal*	

Infinite finite sets def

$\mathbb{Z}$	The set of integers	$\{\ldots, -2, -1, 0, 1, 2, \ldots\}$
$\mathbb{Z}^+$	The set of positive integers	$\{1, 2, \ldots\}$
$\mathbb{N}$	The set of nonnegative integers	$\{0, 1, 2, \ldots\}$
$\mathbb{Q}$	The set of rational numbers	$\left\{\frac{p}{q} \mid p \in \mathbb{Z} \text{ and } q \in \mathbb{Z} \text{ and } q \neq 0\right\}$
$\mathbb{R}$	The set of real numbers	

$$\_\_ \subsetneq \_\_ \subsetneq \_\_ \subsetneq \_\_ \subsetneq \_\_$$

The above sets are all **infinite**.

A **finite** set is one whose distinct elements can be counted by a natural number.

Examples of finite sets: $\emptyset$, $\{\sqrt{2}\}$

Motivating question: Are some of the above sets *bigger than* others?

Musical chairs analogy

Analogy: Musical chairs

People try to sit down when the music stops

Person☼ sits in Chair 1, Person☺ sits in Chair 2,

Person☹ is left standing!

What does this say about the number of chairs and the number of people?

Defined functions

Defining functions A function is defined by its (1) domain, (2) codomain, and (3) rule assigning each element in the domain exactly one element in the codomain. The domain and codomain are nonempty sets. The rule can be depicted as a table, formula, English description, etc.

(Rosen p139)

Example: $f_A : \mathbb{R}^+ \to \mathbb{Q}$ with $f_A(x) = x$ is **not** a well-defined function because

Example: $f_B : \mathbb{Q} \to \mathbb{Z}$ with $f_B\left(\frac{p}{q}\right) = p + q$ is **not** a well-defined function because

Example: $f_C : \mathbb{Z} \to \mathbb{R}$ with $f_C(x) = \frac{x}{|x|}$ is **not** a well-defined function because

Injective cardinality definition

Definition (Rosen p141): A function $f : D \to C$ is **one-to-one** (or injective) means for every a, b in the domain D, if $f(a) = f(b)$ then $a = b$.

Definition: For sets A, B, we say that **the cardinality of A is no bigger than the cardinality of B**, and write $|A| \leq |B|$, to mean there is a one-to-one function with domain A and codomain B.

Injective cardinality musical chairs

In the analogy: The function $sitter : \{Chair1, Chair2\} \to \{Person☼, Person☺, Person☹\}$ given by $sitter(Chair1) = Person☼$, $sitter(Chair2) = Person☺$, is one-to-one and witnesses that

$$|\{Chair1, Chair2\}| \leq |\{Person☼, Person☺, Person☹\}|$$

Rna injective cardinality

Let S_2 be the set of RNA strands of length 2.

Statement	True/False , justification
$\lvert\{\mathtt{A},\mathtt{U},\mathtt{G},\mathtt{C}\}\rvert \leq \lvert S_2\rvert$	
$\lvert\{\mathtt{A},\mathtt{U},\mathtt{G},\mathtt{C}\} \times \{\mathtt{A},\mathtt{U},\mathtt{G},\mathtt{C}\}\rvert \leq \lvert S_2\rvert$	

Surjective cardinality definition

Definition (Rosen p143): A function $f : D \to C$ is **onto** (or surjective) means for every b in the codomain, there is an element a in the domain with $f(a) = b$.

Formally, $f : D \to C$ is onto means _____.

Definition: For sets A, B, we say that **the cardinality of A is no smaller than the cardinality of B**, and write $|A| \geq |B|$, to mean there is an onto function with domain A and codomain B.

Surjective cardinality musical chairs

In the analogy: The function $triedToSit : \{Person☼, Person☺, Person☹\} \to \{Chair1, Chair2\}$ given by $triedToSit(Person☼) = Chair1$, $triedToSit(Person☺) = Chair2$, $triedToSit(Person☹) = Chair2$, is onto and witnesses that

$$|\{Person☼, Person☺, Person☹\}| \geq |\{Chair1, Chair2\}|$$

Rna surjective cardinality

Let S_2 be the set of RNA strands of length 2.

Statement	True/False , justification
$\lvert S_2\rvert \geq \lvert\{\mathtt{A},\mathtt{U},\mathtt{G},\mathtt{C}\}\rvert$	
$\lvert S_2\rvert \geq \lvert\{\mathtt{A},\mathtt{U},\mathtt{G},\mathtt{C}\} \times \{\mathtt{A},\mathtt{U},\mathtt{G},\mathtt{C}\}\rvert$	

Bijection definition

Definition (Rosen p144): A function $f : D \to C$ is a **bijection** means that it is both one-to-one and onto. The **inverse** of a bijection $f : D \to C$ is the function $g : C \to D$ such that $g(b) = a$ iff $f(a) = b$.

For nonempty sets A, B we say

$|A| \leq |B|$ means there is a one-to-one function with domain A, codomain B
$|A| \geq |B|$ means there is an onto function with domain A, codomain B
$|A| = |B|$ means there is a bijection with domain A, codomain B

Cardinality properties

Properties of cardinality

$$\forall A\ (\ |A| = |A|\ )$$
$$\forall A\ \forall B\ (\ |A| = |B|\ \rightarrow\ |B| = |A|\ )$$
$$\forall A\ \forall B\ \forall C\ (\ (|A| = |B|\ \wedge\ |B| = |C|)\ \rightarrow\ |A| = |C|\ )$$

Extra practice with proofs: Use the definitions of bijections to prove these properties.

Cantor schroder bernstein theorem

Cantor-Schroder-Bernstein Theorem: For all nonempty sets,

$$|A| = |B| \quad \text{if and only if} \quad (|A| \leq |B| \text{ and } |B| \leq |A|) \quad \text{if and only if} \quad (|A| \geq |B| \text{ and } |B| \geq |A|)$$

To prove $|A| = |B|$, we can do any **one** of the following

- Prove there exists a bijection $f : A \to B$;
- Prove there exists a bijection $f : B \to A$;
- Prove there exists two functions $f_1 : A \to B$, $f_2 : B \to A$ where each of f_1, f_2 is one-to-one.
- Prove there exists two functions $f_1 : A \to B$, $f_2 : B \to A$ where each of f_1, f_2 is onto.

Logical equivalence

Logical equivalence	Two compound propositions are **logically equivalent** means that they have the same truth values for all settings of truth values to their propositional variables.
Tautology	A compound proposition that evaluates to true for all settings of truth values to its propositional variables; it is abbreviated T.
Contradiction	A compound proposition that evaluates to false for all settings of truth values to its propositional variables; it is abbreviated F.
Contingency	A compound proposition that is neither a tautology nor a contradiction.

Logical equivalence extra note

Can replace p and q with any compound proposition

Consistency def

Definition: A collection of compound propositions is called **consistent** if there is an assignment of truth values to the propositional variables that makes each of the compound propositions true.

Consistency example

Consistency:

> Whenever the system software is being upgraded, users cannot access the file system. If users can access the file system, then they can save new files. If users cannot save new files, then the system software is not being upgraded.

1. Translate to symbolic compound propositions
2. Look for some truth assignment to the propositional variables for which all the compound propositions output T

Redundancy algorithm

Consider the following algorithm to introduce redundancy in a string of 0s and 1s.

Create redundancy by repeating each bit three times

```
1  procedure redun3(a_{k-1} ··· a_0: a binary string)
2  for i := 0 to k − 1
3    c_{3i} := a_i
4    c_{3i+1} := a_i
5    c_{3i+2} := a_i
6  return c_{3k-1} ··· c_0
```

Decode sequence of bits using majority rule on consecutive three bit sequences

```
1  procedure decode3(c_{3k-1} ··· c_0: a binary string whose length is an integer multiple of 3)
2  for i := 0 to k − 1
3    if exactly two or three of c_{3i}, c_{3i+1}, c_{3i+2} are set to 1
4      a_i := 1
5    else
6      a_i := 0
7  return a_{k-1} ··· a_0
```

Give a recursive definition of the set of outputs of the *redun*3 procedure, *Out*,

Basis step: _____

Recursive step: _____

Consider the message $m = 0001$ so that the sender calculates $redun3(m) = redun3(0001) = 000000000111$.

Introduce ___ errors into the message so that the signal received by the receiver is _____ but the receiver is still able to decode the original message.
Challenge: what is the biggest number of errors you can introduce?

Building a circuit for line 3 in *decode* procedure: given three input bits, we need to determine whether the majority is a 0 or a 1.

c_{3i}	c_{3i+1}	c_{3i+2}	a_i
1	1	1	
1	1	0	
1	0	1	
1	0	0	
0	1	1	
0	1	0	
0	0	1	
0	0	0	

Circuit

Predicate definition

Definition: A **predicate** is a function from a given set (domain) to $\{T, F\}$.

A predicate can be applied, or **evaluated** at, an element of the domain.

Predicate equivalent definition

Two predicates over the same domain are **equivalent** means they evaluate to the same truth values for all possible assignments of domain elements to the input.

Predicate truth tables

Input	Output		
	$P(x)$	$N(x)$	$Mystery(x)$
x	$[x]_{2c,3} > 0$	$[x]_{2c,3} < 0$	
000	F		T
001	T		T
010	T		T
011	T		F
100	F		F
101	F		T
110	F		F
111	F		T

The domain for each of the predicates $P(x)$, $N(x)$, $Mystery(x)$ is _____.

Truth set definition

Definition: The **truth set** of a predicate is the collection of all elements in its domain where the predicate evaluates to T.

Truth set exercise

The truth set for the predicate $P(x)$ is _____.

The truth set for the predicate $N(x)$ is _____.

The truth set for the predicate $Mystery(x)$ is _____.

Quantification definition

Definitions (Rosen 40-45):
The **universal quantification** of $P(x)$ is the statement "$P(x)$ for all values of x in the domain" and is written $\forall x P(x)$. An element for which $P(x) = F$ is called a **counterexample** of $\forall x P(x)$.

The **existential quantification** of $P(x)$ is the statement "There exists an element x in the domain such that $P(x)$" and is written $\exists x P(x)$. An element for which $P(x) = T$ is called a **witness** of $\exists x P(x)$.

Example: _____ is a true existential quantification.

Quantification logical equivalence

Statements involving predicates and quantifiers are **logically equivalent** means they have the same truth value no matter which predicates (domains and functions) are substituted in.

Quantifier version of De Morgan's laws: $\boxed{\neg\forall x P(x) \equiv \exists x\,(\neg P(x))}$ $\boxed{\neg\exists x Q(x) \equiv \forall x\,(\neg Q(x))}$

Example: _____ is a false universal quantification. It is logically equivalent to _____

Rna strand recall

Recall: Each RNA strand is a string whose symbols are elements of the set $B = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{U}\}$. The **set of all RNA strands** is called S. The function *rnalen* that computes the length of RNA strands in S is:

		$rnalen : S$	$\to \mathbb{Z}^+$
Basis Step:	If $b \in B$ then	$rnalen(b)$	$= 1$
Recursive Step:	If $s \in S$ and $b \in B$, then	$rnalen(sb)$	$= 1 + rnalen(s)$

Predicate rna example

Example predicates on S

$H(s) = T$	Truth set of H is _____
$L_3(s) = \begin{cases} T & \text{if } rnalen(s) = 3 \\ F & \text{otherwise} \end{cases}$	Strand where L_3 evaluates to T is e.g._____ Strand where L_3 evaluates to F is e.g. _____
$F_{\mathtt{A}}$ is defined recursively by: Basis step: $F_{\mathtt{A}}(\mathtt{A}) = T$, $F_{\mathtt{A}}(\mathtt{C}) = F_{\mathtt{A}}(\mathtt{G}) = F_{\mathtt{A}}(\mathtt{U}) = F$ Recursive step: If $s \in S$ and $b \in B$, then $F_{\mathtt{A}}(sb) = F_{\mathtt{A}}(s)$	Strand where $F_{\mathtt{A}}$ evaluates to T is e.g._____ Strand where $F_{\mathtt{A}}$ evaluates to F is e.g. _____
$P_{\mathtt{AUC}}$ is defined as the predicate whose truth set is the collection of RNA strands where the string `AUC` is a substring (appears inside s, in order and consecutively)	Strand where $P_{\mathtt{AUC}}$ evaluates to T is e.g._____ Strand where $P_{\mathtt{AUC}}$ evaluates to F is e.g. _____

Cartesian product definition

Definition (Rosen p123): The **Cartesian product** of the sets A and B, $A \times B$, is the set of all ordered pairs (a, b), where $a \in A$ and $b \in B$. That is: $A \times B = \{(a, b) \mid (a \in A) \wedge (b \in B)\}$. The Cartesian product of the sets $A_1, A_2, \ldots, A_n$, denoted by $A_1 \times A_2 \times \cdots \times A_n$, is the set of ordered n-tuples $(a_1, a_2, ..., a_n)$, where a_i belongs to A_i for $i = 1, 2, \ldots, n$. That is, $A_1 \times A_2 \times \cdots \times A_n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A_i \text{ for } i = 1, 2, \ldots, n\}$

Rna strand recall

Recall: Each RNA strand is a string whose symbols are elements of the set $B = \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\}$. The **set of all RNA strands** is called S. The function *rnalen* that computes the length of RNA strands in S is:

		$rnalen : S$	$\to \mathbb{Z}^+$
Basis Step:	If $b \in B$ then	$rnalen(b)$	$= 1$
Recursive Step:	If $s \in S$ and $b \in B$, then	$rnalen(sb)$	$= 1 + rnalen(s)$

Rna basecount example

A function *basecount* that computes the number of a given base b appearing in a RNA strand s is:

		$basecount : S \times B$	$\to \mathbb{N}$
Basis Step:	If $b_1 \in B, b_2 \in B$	$basecount(b_1, b_2)$	$= \begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases}$
Recursive Step:	If $s \in S, b_1 \in B, b_2 \in B$	$basecount(sb_1, b_2)$	$= \begin{cases} 1 + basecount(s, b_2) & \text{when } b_1 = b_2 \\ basecount(s, b_2) & \text{when } b_1 \neq b_2 \end{cases}$

L with domain $S \times \mathbb{Z}^+$ is defined by, for $s \in S$ and $n \in \mathbb{Z}^+$,

$$L(s, n) = \begin{cases} T & \text{if } rnalen(s) = n \\ F & \text{otherwise} \end{cases}$$

Element where L evaluates to T: _____

Element where L evaluates to F: _____

BC with domain _____ is defined by, for $s \in S$ and $b \in B$ and $n \in \mathbb{N}$,

$$BC(s, b, n) = \begin{cases} T & \text{if } basecount(s, b) = n \\ F & \text{otherwise} \end{cases}$$

Element where BC evaluates to T: _____

Element where BC evaluates to F: _____

Predicate notation

Notation: for a predicate P with domain $X_1 \times \cdots \times X_n$ and a n-tuple $(x_1, \ldots, x_n)$ with each $x_i \in X$, we write $P(x_1, \ldots, x_n)$ to mean $P(\ (x_1, \ldots, x_n)\ )$.

Rna basecount example two

$\exists t\ BC(t)$ In English: _____

Witness that proves this existential quantification is true: _____

$\forall(s, b, n)\ (\ BC(s, b, n)\ )$ In English: _____

Counterexample that proves this universal quantification is false: _____

New predicates from old $BC(s, b, n)$ means $basecount(s, b) = n$.

Predicate	Domain	Example domain element where predicate is T
$basecount(s, b) = 3$		
$basecount(s, \texttt{A}) = n$		
$\exists n \in \mathbb{N}\ (basecount(s, b) = n)$		
$\forall b \in B\ (basecount(s, b) = 1)$		

Alternating quantifiers

Alternating quantifiers

$$\forall s \exists n \; BC(s, \texttt{A}, n)$$

In English: _____

$$\exists n \forall s \; BC(s, \texttt{U}, n)$$

In English: _____

Evaluate each quantified statement as T or F.

$\forall s \; \forall b \; \exists n \; BC(s, b, n)$	$\forall s \; \forall n \; \exists b \; BC(s, b, n)$	$\forall b \; \forall n \; \exists s \; BC(s, b, n)$
$\exists s \; \forall b \; \exists n \; BC(s, b, n)$	$\forall s \; \exists n \; \forall b \; BC(s, b, n)$	$\exists b \; \exists n \; \forall s \; BC(s, b, n)$

Extra example: Write the negation of each of the statements above, and use De Morgan's law to find a logically equivalent version where the negation is applied only to the BC predicate (not next to a quantifier).

Making change example

For which nonnegative integers n can we make change for n with coins of value 5 cents and 3 cents?

Restating: We can make change for _____, we cannot make change for _____, and

_____*

Strong induction def

New! Proof by Strong Induction (Rosen 5.2 p337)
To prove that a universal quantification over the set of all integers greater than or equal to some base integer b holds, pick a fixed nonnegative integer j and then:

Basis Step:	Show the statement holds for b, $b+1$, …, $b+j$.
Recursive Step:	Consider an arbitrary integer n greater than or equal to $b+j$, assume (as the **strong induction hypothesis**) that the property holds for **each of** b, $b+1$, …, n, and use this and other facts to prove that the property holds for $n+1$.

$\mathbb{N}$	The set of natural numbers	$\{0,1,2,3,\ldots\}$
$\mathbb{Z}^{\geq b}$	The set of integers greater than or equal a basis element b	$\{b, b+1, b+2, b+3, \ldots\}$

Making change proof two ways

Proof of $\star$ by mathematical induction ($b = 8$)

Basis step: WTS property is true about 8

Recursive step: Consider an arbitrary $n \geq 8$. Assume (as the IH) that there are nonnegative integers x, y such that $n = 5x + 3y$. WTS that there are nonnegative integers x', y' such that $n + 1 = 5x' + 3y'$. We consider two cases, depending on whether any 5 cent coins are used for n.

Case 1: Assume $x \geq 1$. Define $x' = x - 1$ and $y' = y + 2$ (both in $\mathbb{N}$ by case assumption). Calculating:

$$5x' + 3y' \overset{\text{by def}}{=} 5(x-1) + 3(y+2) = 5x - 5 + 3y + 6$$
$$\overset{\text{rearranging}}{=} (5x + 3y) - 5 + 6$$
$$\overset{\text{IH}}{=} n - 5 + 6 = n + 1$$

Case 2: Assume $x = 0$. Therefore $n = 3y$, so since $n \geq 8$, $y \geq 3$. Define $x' = 2$ and $y' = y - 3$ (both in $\mathbb{N}$ by case assumption). Calculating:

$$5x' + 3y' \overset{\text{by def}}{=} 5(2) + 3(y-3) = 10 + 3y - 9$$
$$\overset{\text{rearranging}}{=} 3y + 10 - 9$$
$$\overset{\text{IH and case}}{=} n + 10 - 9 = n + 1$$

Proof of $\star$ by strong induction ($b = 8$ and $j = 2$)

Basis step: WTS property is true about $8, 9, 10$

Recursive step: Consider an arbitrary $n \geq 10$. Assume (as the IH) that the property is true about each of $8, 9, 10, \ldots, n$. WTS that there are nonnegative integers x', y' such that $n + 1 = 5x' + 3y'$.

Binary expansions exist proof

Representing positive integers

Theorem: Every positive integer is a sum of (one or more) distinct powers of 2. *binary expansions exist!*

Proof by strong induction, with $b = 1$ and $j = 0$.

Basis step: WTS property is true about 1.

Recursive step: Consider an arbitrary integer $n \geq 1$. Assume (as the IH) that the property is true about each of $1, \ldots, n$. WTS that the property is true about $n + 1$.

Strong induction def

New! Proof by Strong Induction (Rosen 5.2 p337)
To prove that a universal quantification over the set of all integers greater than or equal to some base integer b holds, pick a fixed nonnegative integer j and then:

Basis Step:	Show the statement holds for b, $b + 1$, $\ldots$, $b + j$.
Recursive Step:	Consider an arbitrary integer n greater than or equal to $b + j$, assume (as the **strong induction hypothesis**) that the property holds for **each of** b, $b + 1$, $\ldots$, n, and use this and other facts to prove that the property holds for $n + 1$.

$\mathbb{N}$	The set of natural numbers	$\{0, 1, 2, 3, \ldots\}$
$\mathbb{Z}^{\geq b}$	The set of integers greater than or equal a basis element b	$\{b, b + 1, b + 2, b + 3, \ldots\}$

Prime number def

Definition: An integer p greater than 1 is called **prime** if the only positive factors of p are 1 and p. A positive integer that is greater than 1 is not prime is called composite.

Fundamental theorem proof

Theorem: Every positive integer *greater than 1* is a product of (one or more) primes.

Proof by strong induction, with $b = 2$ and $j = 0$.

Basis step: WTS property is true about 2.

Recursive step: Consider an arbitrary integer $n \geq 2$. Assume (as the IH) that the property is true about each of $2, \ldots, n$. WTS that the property is true about $n + 1$.

Case 1:

Case 2:

Prime number def

Definition: An integer p greater than 1 is called **prime** if the only positive factors of p are 1 and p. A positive integer that is greater than 1 and is not prime is called composite.

Proof by contradiction def

New! Proof by Contradiction
To prove that a statement p is true, pick another statement r and once we show that $\neg p \to (r \wedge \neg r)$ then we can conclude that p is true.
Informally The statement we care about can't possible be false, so it must be true.

Least greatest proofs

Prove or **disprove**: There is a least prime number.

Prove or **disprove**: There is a greatest integer.

Approach 1, De Morgan's and universal generalization:

Approach 2, proof by contradiction:

Extra examples: Prove or disprove that $\mathbb{N}$, $\mathbb{Q}$ each have a least and a greatest element. Prove that there is no greatest prime number.

Rational number def

The **set of rational numbers**, $\mathbb{Q}$ is defined as

$$\left\{ \frac{p}{q} \mid p \in \mathbb{Z} \text{ and } q \in \mathbb{Z} \text{ and } q \neq 0 \right\} \quad \text{or, equivalently,} \quad \{x \in \mathbb{R} \mid \exists p \in \mathbb{Z} \exists q \in \mathbb{Z}^+ (p = x \cdot q)\}$$

Extra practice: Use the definition of set equality to prove that the definitions above give the same set.

Proof square root2

Goal: The square root of 2 is not a rational number. In other words: $\neg \exists x \in \mathbb{Q}(x^2 - 2 = 0)$

Attempted proof: The definition of the set of rational numbers is the collection of fractions p/q where p is an integer and q is a nonzero integer. Looking for a **witness** p and q, we can write the square root of 2 as the fraction $\sqrt{2}/1$, where 1 is a nonzero integer. Since the numerator is not in the domain, this witness is not allowed, and we have shown that the square root of 2 is not a fraction of integers (with nonzero denominator). Thus, the square root of 2 is not rational.

The problem in the above attempted proof is that _____

Proof:

Lemma 1: For every two integers p and q, not both zero, $gcd\left(\frac{p}{gcd(p,q)}, \frac{q}{gcd(p,q)}\right) = 1$.

Lemma 2: For every two integers a and b, not both zero, with $gcd(a, b) = 1$, it is not the case that both a is even and b is even.

Lemma 3: For every integer x, x is even if and only if x^2 is even.

Gcd def

Greatest common divisor Let a and b be integers, not both zero. The largest integer d such that d is a factor of a and d is a factor of b is called the greatest common divisor of a and b and is denoted by $gcd(a, b)$.

Proposition def

Proposition	Declarative sentence that is true or false (not both).
Propositional variable	Variable that represents a proposition.
Compound proposition	New propositions formed from existing propositions (potentially) using logical operators.
Truth table	Table with 1 row for each of the possible combinations of truth values of the input and an additional column that shows the truth value of the result of the operation corresponding to a particular row.

Note: A propositional variable is one example of a compound proposition.

Logical operators

Logical operators aka propositional connectives

Conjunction	AND	$\wedge$	`\land`	2 inputs	Evaluates to T when **both** inputs are T
Exclusive or	XOR	$\oplus$	`\oplus`	2 inputs	Evaluates to T when **exactly one** of inputs is T
Disjunction	OR	$\vee$	`\lor`	2 inputs	Evaluates to T when **at least one** of inputs is T
Negation	NOT	$\neg$	`\lnot`	1 input	Evaluates to T when its input is F

Logical operators truth tables

Input		Output		
		Conjunction	**Exclusive or**	**Disjunction**
p	q	$p \wedge q$	$p \oplus q$	$p \vee q$
T	T	T	F	T
T	F	F	T	T
F	T	F	T	T
F	F	F	F	F

Input	Output
	Negation
p	$\neg p$
T	F
F	T

Logical operators exercise

Input			Output	
p	q	r	$(p \wedge q) \oplus (\ (p \oplus q) \wedge r\ )$	$(p \wedge q) \vee (\ (p \oplus q) \wedge r\ )$
T	T	T		
T	T	F		
T	F	T		
T	F	F		
F	T	T		
F	T	F		
F	F	T		
F	F	F		

Logical equivalence

Logical equivalence	Two compound propositions are **logically equivalent** means that they have the same truth values for all settings of truth values to their propositional variables.
Tautology	A compound proposition that evaluates to true for all settings of truth values to its propositional variables; it is abbreviated T.
Contradiction	A compound proposition that evaluates to false for all settings of truth values to its propositional variables; it is abbreviated F.
Contingency	A compound proposition that is neither a tautology nor a contradiction.

Logical equivalence exercise

Extra Example: Which of the compound propositions in the table below are logically equivalent?

Input		Output				
p	q	$\neg(p \wedge \neg q)$	$\neg(\neg p \vee \neg q)$	$(\neg p \vee q)$	$(\neg q \vee \neg p)$	$(p \wedge q)$
T	T					
T	F					
F	T					
F	F					

Logical equivalence example

(Some) logical equivalences) cf. Rosen pp. 26-28

$p \vee q \equiv q \vee p$	$p \wedge q \equiv q \wedge p$	**Commutativity** Ordering of terms
$(p \vee q) \vee r \equiv p \vee (q \vee r)$	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	**Associativity** Grouping of terms
$p \wedge F \equiv F \qquad p \vee T \equiv T$	$p \wedge T \equiv p \qquad p \vee F \equiv p$	**Absorption** aka short circuit evaluation
$\neg(p \wedge q) \equiv \neg p \vee \neg q$	$\neg(p \vee q) \equiv \neg p \wedge \neg q$	**DeMorgan's Laws**

Can replace p and q with any compound proposition

Truth table to compound proposition application

Application: design a circuit given a desired input-output relationship.

Input		Output	
p	q	$mystery_1$	$mystery_2$
T	T	T	F
T	F	T	F
F	T	F	F
F	F	T	T

Input			Output
p	q	r	?
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

A compound proposition that gives output $mystery_1$ is: _____

A compound proposition that gives output $mystery_2$ is: _____

Dnf cnf definition

Definition A compound proposition is in **disjunctive normal form** (DNF) means that it is an OR of ANDs of variables and their negations.

Definition A compound proposition is in **conjunctive normal form** (CNF) means that it is an AND of ORs of variables and their negations.

Extra example: A compound proposition that gives output ? is:

Hypothesis conclusion

The only way to make the conditional statement $p \to q$ false is to _____

The **hypothesis** of $p \to q$ is _____ The **antecedent** of $p \to q$ is _____

The **conclusion** of $p \to q$ is _____ The **consequent** of $p \to q$ is _____

Logical equivalence example two

Input		Output				
		Conjunction	Exclusive or	Disjunction	Conditional	Biconditional
p	q	$p \wedge q$	$p \oplus q$	$p \vee q$	$p \to q$	$p \leftrightarrow q$
T	T	T	F	T	T	T
T	F	F	T	T	F	F
F	T	F	T	T	T	F
F	F	F	F	F	T	T

Examples

$p \to q \equiv \neg p \vee q$ because _____

$p \leftrightarrow q$ is not logically equivalent to $p \wedge q$ because _____

$\neg(p \leftrightarrow q) \equiv p \oplus q$ because _____

$p \to q$ is not logically equivalent to $q \to p$ because _____

$p \leftrightarrow q \equiv q \leftrightarrow p$ because _____

Converse inverse contrapositive

The **converse** of $p \to q$ is _____

The **inverse** of $p \to q$ is _____ Which of these is logically equivalent to $p \to q$?

The **contrapositive** of $p \to q$ is _____

Compound propositions translation

Translation: Express each of the following sentences as compound propositions, using the given propositions.

"A sufficient condition for the warranty to be good is that you bought the computer less than a year ago"	w is "the warranty is good" b is "you bought the computer less than a year ago"

"Whenever the message was sent from an unknown system, it is scanned for viruses."	s is "The message is scanned for viruses" u is "The message was sent from an unknown system"

"I will complete my to-do list only if I put a reminder in my calendar"	r is "I will complete my to-do list" c is "I put a reminder in my calendar"