AI Concepts, Python, and GitHub

Lecture Outline

- Fundamental Concepts
- Knowledge Representation
- Basics of Logic
  - Syllogistic Logic
- Overview of Reasoning
  - Deductive, Inductive, Abductive
- Learning Python Coding
- Coding Help
- Git and GitHub

Big Picture

- One of the hallmarks of intelligence is the ability to reason
- Reasoning allows us to solve problems
- If we want to build intelligent machines, we must be able to automate reasoning

Big Picture

- One of the hallmarks of intelligence is the ability to reason
- Reasoning allows us to solve problems
- If we want to build intelligent machines, we must be able to automate reasoning
- Logic is the study of how we (correctly) reason
- Goals of Logic:
  - Represent knowledge of the world
  - Reason with that knowledge

Knowledge

- Definition of knowledge - ongoing debate among philosophers

- For our purposes, defined as:
  - Statements: Facts, information, descriptions
  - Regarded as true/correct or at least highly probable to be true

Knowledge

- Definition of knowledge - ongoing debate among philosophers

- For our purposes, defined as:
  - Statements: Facts, information, descriptions
  - Regarded as true/correct or at least highly probable to be true

- E.g.

A **dime** is better than a **nickel.**

A **nickel** is better than a **penny.**

Knowledge

- Definition of knowledge - ongoing debate among philosophers

- For our purposes, defined as:
  - Statements: Facts, information, descriptions
  - Regarded as true/correct or at least highly probable to be true

- E.g.

A **dime** is better than a **nickel.**

A **nickel** is better than a **penny.**

- Knowledge acquired through experience or education
  - Perceiving, discovering, or learning

Knowledge

- Definition of knowledge - ongoing debate among philosophers

- For our purposes, defined as:
  - Statements: Facts, information, descriptions
  - Regarded as true/correct or at least highly probable to be true

- E.g.

A **dime** is better than a **nickel.**

A **nickel** is better than a **penny.**

- Knowledge acquired through experience or education
  - Perceiving, discovering, or learning

- Simple enough, but:
  - Communicating?
  - Manipulating knowledge to draw conclusions and derive new knowledge?

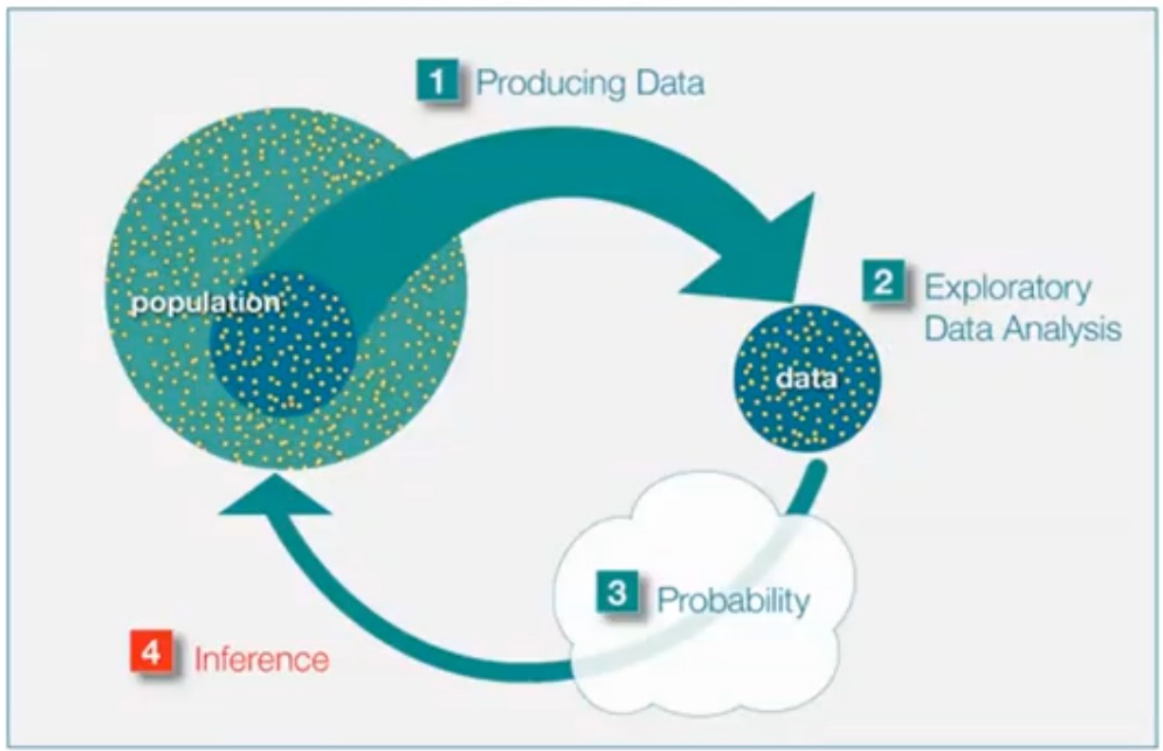
Inference

- A conclusion reached on the basis of evidence and reasoning

Inference

- A conclusion reached on the basis of evidence and reasoning

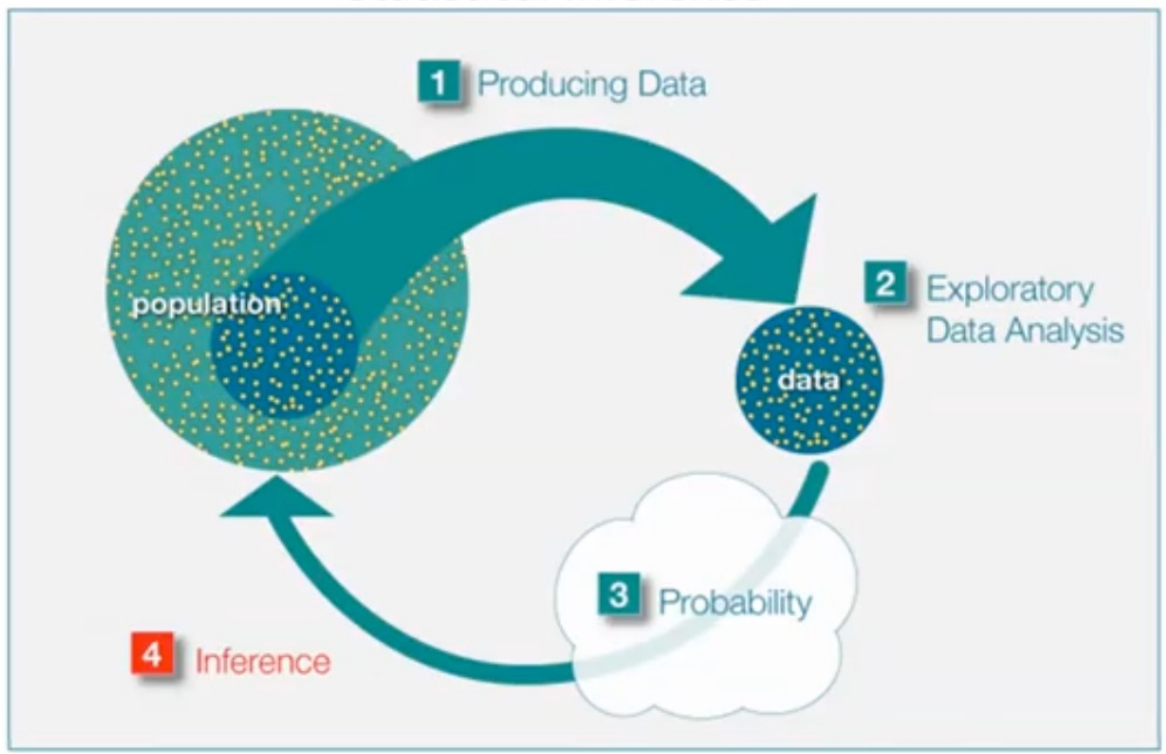
Statistical Inference



Inference

- A conclusion reached on the basis of evidence and reasoning

Statistical Inference

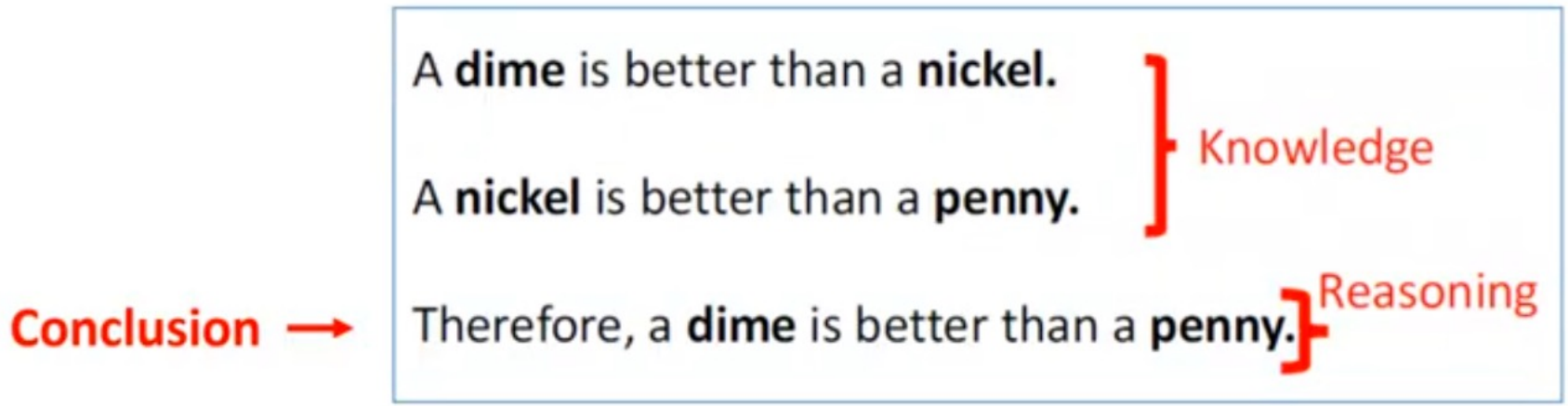


Reasoning

- The drawing of conclusions (inferences) from known or assumed knowledge.

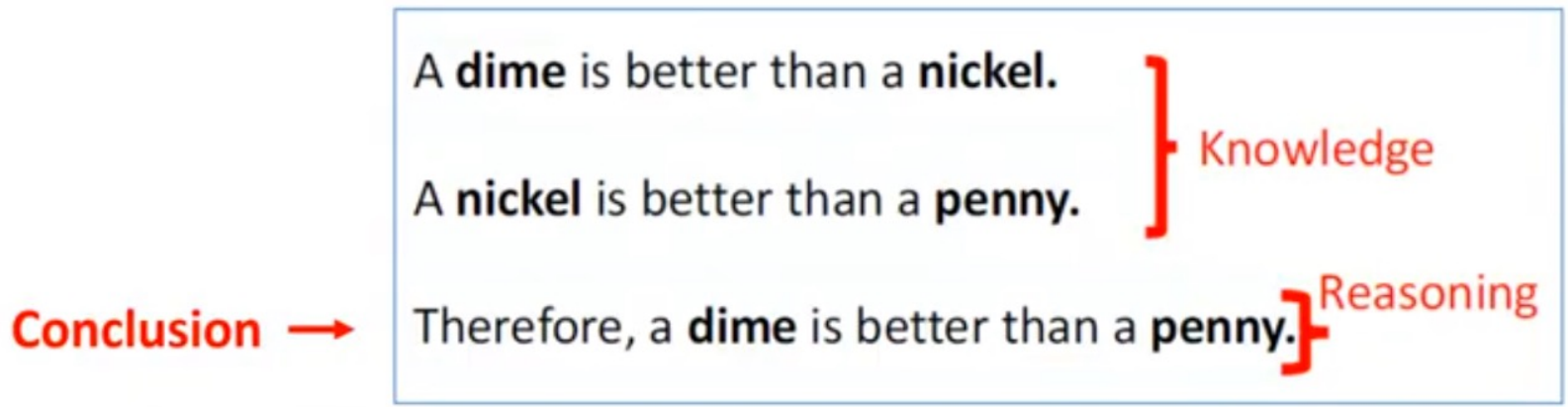
Reasoning

- The drawing of conclusions (inferences) from known or assumed knowledge.



Reasoning

- The drawing of conclusions (inferences) from known or assumed knowledge.



- There are many ways to reason

- Above is an example of logical deductive reasoning
  - i.e. Syllogism

Knowledge Representation

Reasoning with ‘Natural language’

- Problematic:

A **penny** is better than a **nothing.**

Nothing is better than **world peace.**

Knowledge

Invalid Conclusion → Therefore, a **penny** is better than **world peace.**

Reasoning with ‘Natural language’

- Problematic:

A **penny** is better than a **nothing.**

Nothing is better than **world peace.**

Knowledge

Invalid Conclusion → Therefore, a **penny** is better than **world peace.**

- Natural language is tricky (often ambiguous)!
  - An impractically huge knowledge base may be needed to understand context

Reasoning with ‘Natural language’

- Problematic:

A **penny** is better than a **nothing.**

Nothing is better than **world peace.**

Knowledge

Invalid Conclusion → Therefore, a **penny** is better than **world peace.**

- Natural language is tricky (often ambiguous)!
  - An impractically huge knowledge base may be needed to understand context

- We need a more consistent, controlled way to represent knowledge to ensure:
  - Our conclusions are valid
  - A computing system can make effective use of knowledge

Knowledge Representation

- Solving problems in a particular domain often requires knowledge of the objects and how to reason in that domain

Knowledge Representation

- Solving problems in a particular domain often requires knowledge of the objects and how to reason in that domain

- Formal Representation:
  - Standardizes objects and reasoning
  - Example:
    - Logic
    - Programming languages
    - Ontologies

Knowledge Representation

- Solving problems in a particular domain often requires knowledge of the objects and how to reason in that domain

- Formal Representation:
  - Standardizes objects and reasoning
  - Example:
    - Logic
    - Programming languages
    - Ontologies

- Mapping is needed to relate natural to formal

Knowledge Representation

- Solving problems in a particular domain often requires knowledge of the objects and how to reason in that domain

- Formal Representation:
  - Standardizes objects and reasoning
  - Example:
    - Logic
    - Programming languages
    - Ontologies

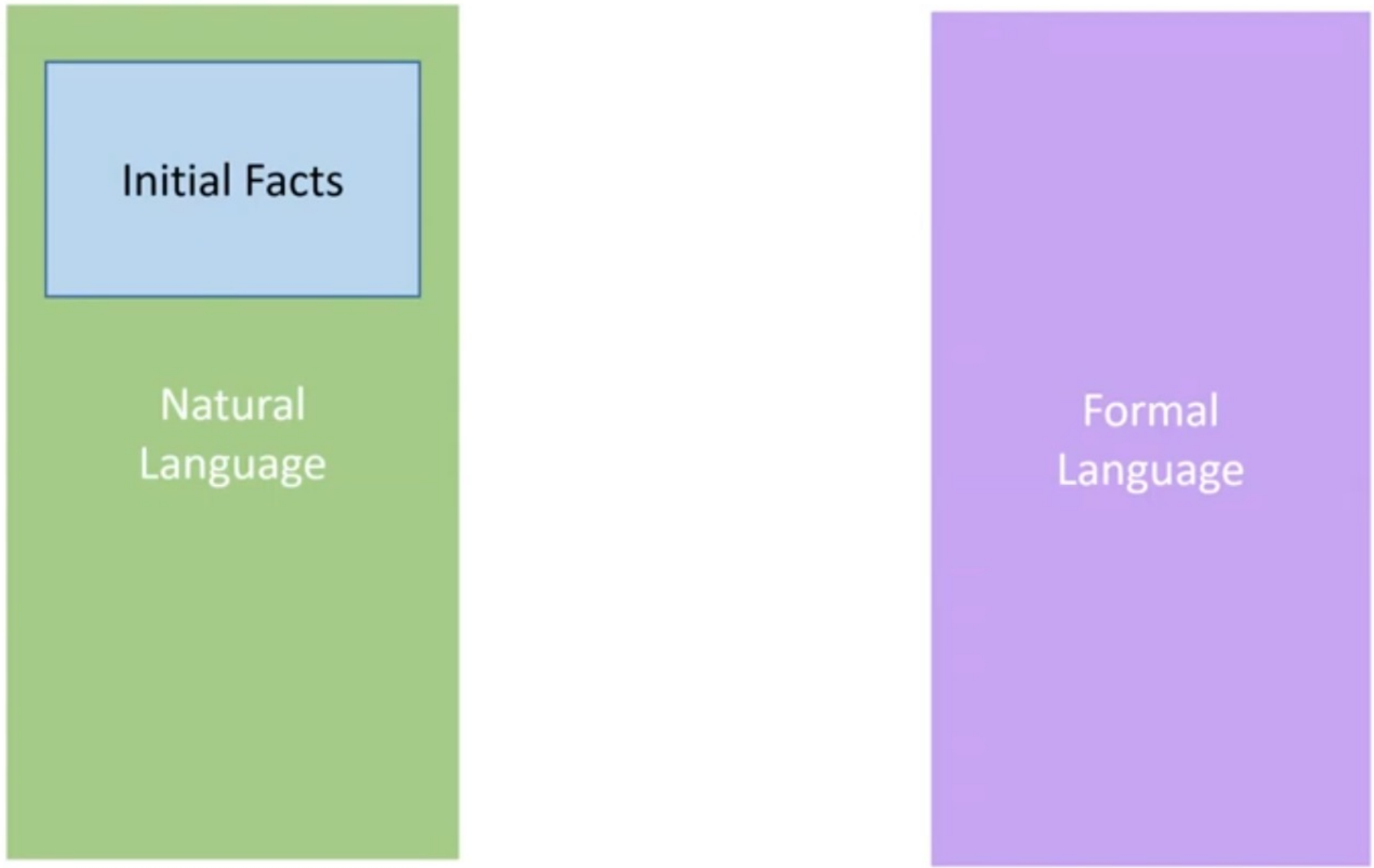
- Mapping is needed to relate natural to formal

- Consequence:
  - Less flexibility

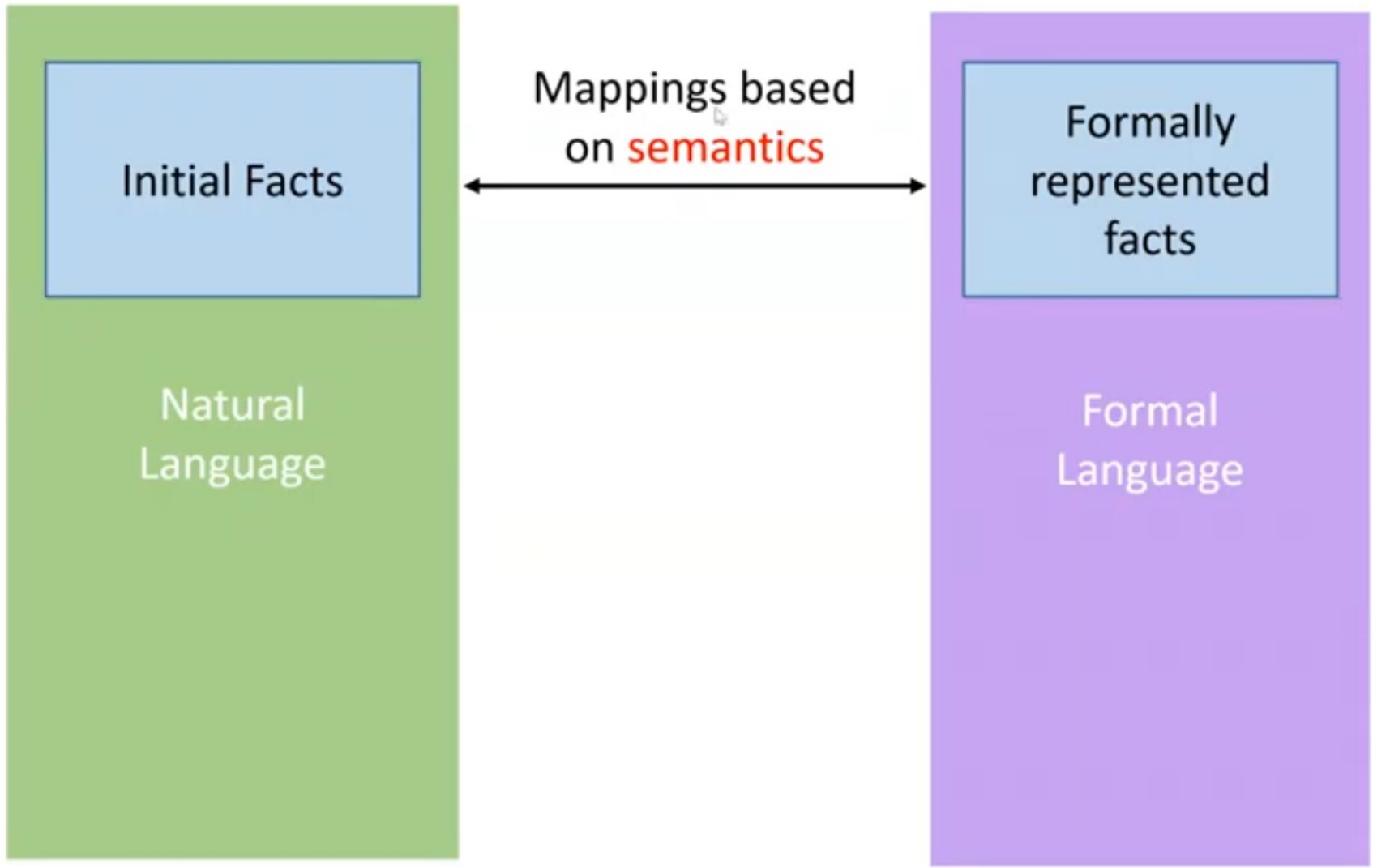
- *"The intended role of knowledge representation in AI is to reduce problems of intelligent action to search problems."*

– Ginsberg (1993)

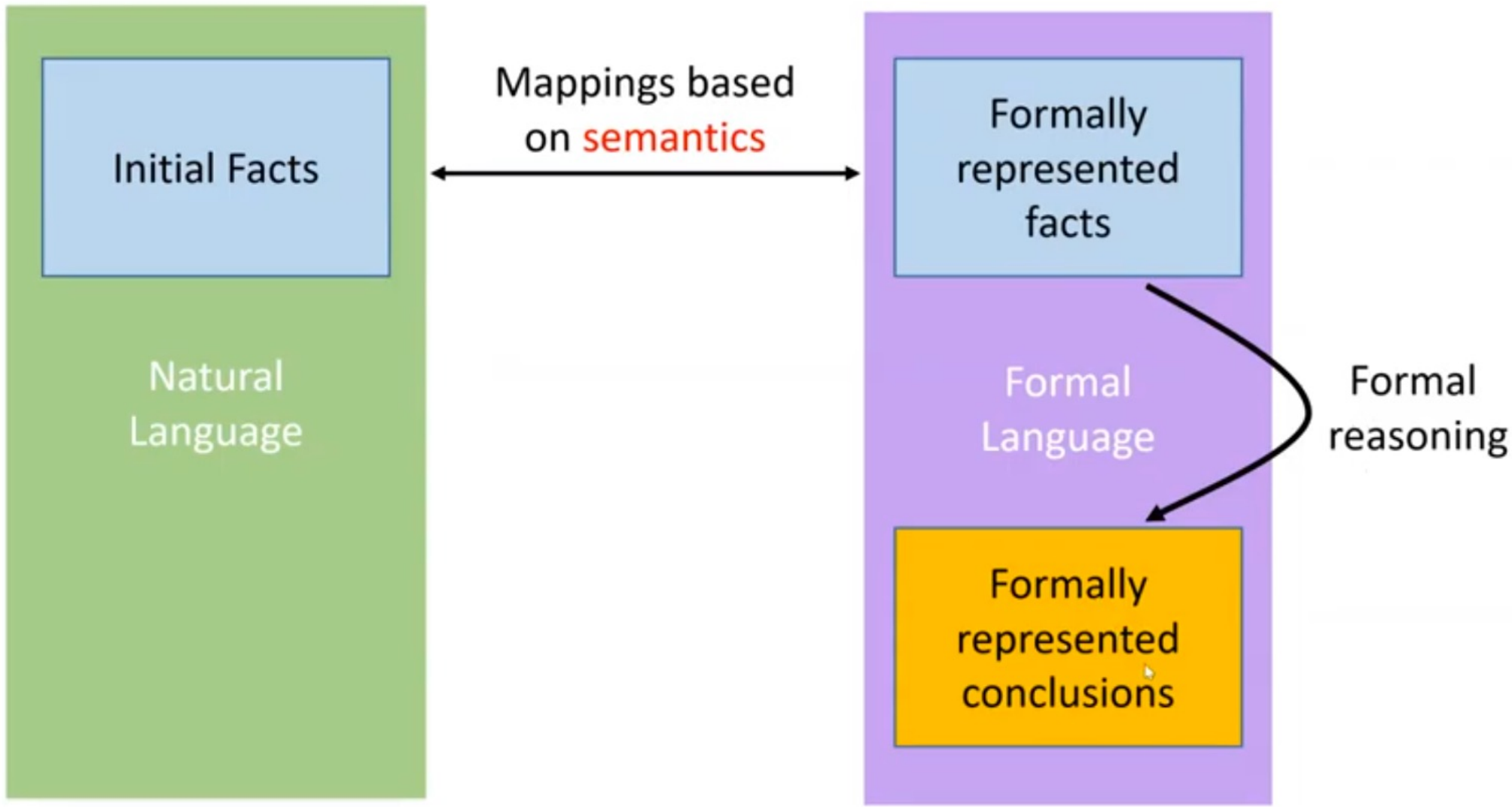
Representation and Mappings



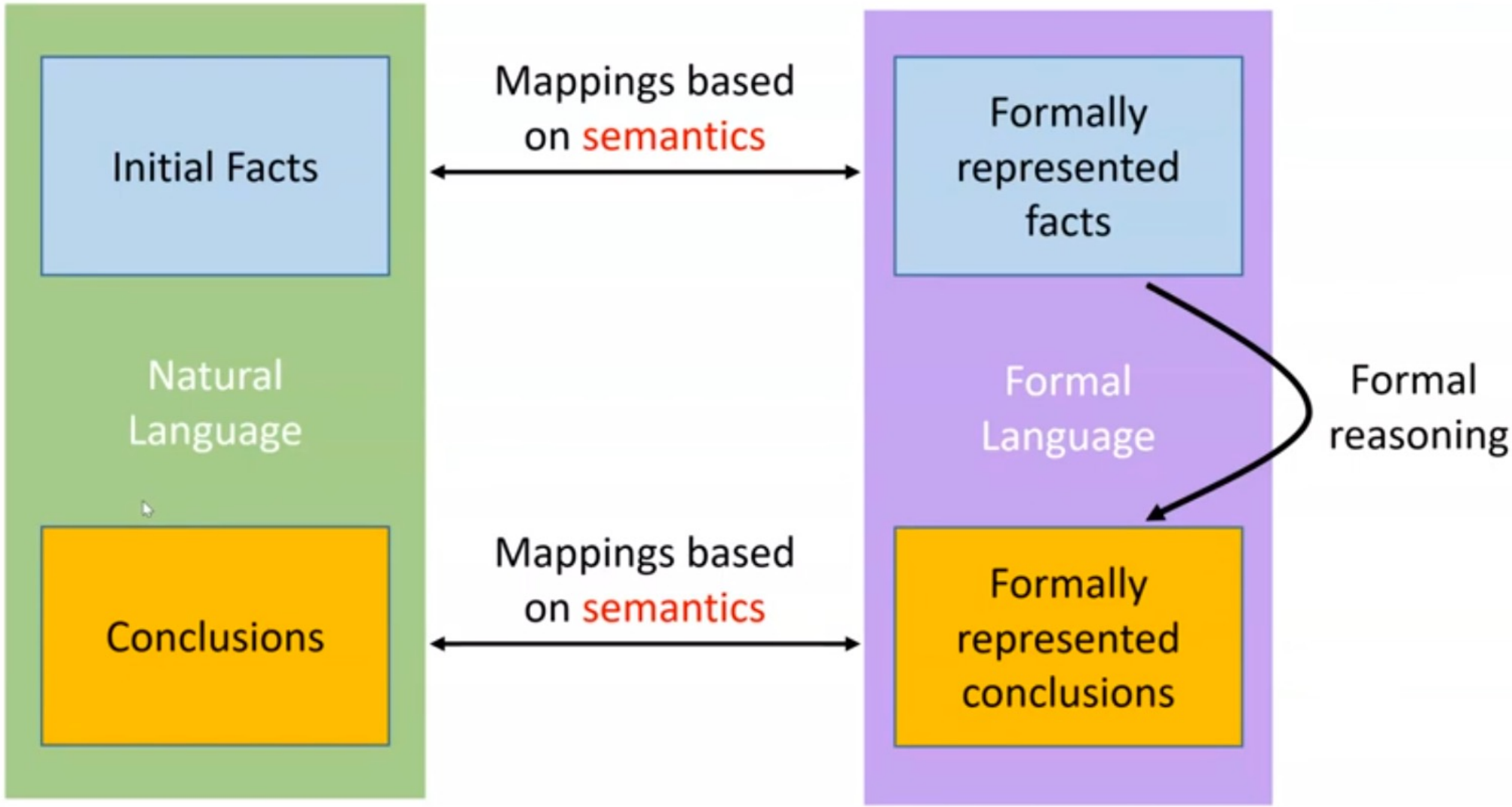
Representation and Mappings



Representation and Mappings



Representation and Mappings



Representation: Symbol Structures

- Knowledge must be represented:
  - Effectively, consistently and in a meaningful way
- Symbol structures - representing bits of knowledge
  - Example:
    - Symbol ‘blue’ denotes a particular color
    - Symbol structure blue(Ryan-car) to denote the fact that my car is blue

Representation: Symbol Structures

- Knowledge must be represented:
  - Effectively, consistently and in a meaningful way
- Symbol structures - representing bits of knowledge
  - Example:
    - Symbol ‘blue’ denotes a particular color
    - Symbol structure blue(Ryan-car) to denote the fact that my car is blue
  - Could be any physical medium

Punch card symbol structure

Representation: Symbol Structures

- Knowledge must be represented:
  - Effectively, consistently and in a meaningful way

- Symbol structures - representing bits of knowledge
  - Example:
    - Symbol ‘blue’ denotes a particular color
    - Symbol structure blue(Ryan-car) to denote the fact that my car is blue

  - Could be any physical medium
  - Computers make this easier
    - Facts: data structures (list, dictionary, array, dataframe)
    - Reasoning: program code

Punch card symbol structure

Programming languages

Knowledge Representation Requirements

- Representational adequacy:
  - Represent all knowledge you need to reason with

Knowledge Representation Requirements

- Representational adequacy:
  - Represent all knowledge you need to reason with
- Inferential adequacy:
  - Allow new knowledge to be inferred from a basic set of facts
    - Don’t innumerate negatives that could be covered by one positive

Knowledge Representation Requirements

- Representational adequacy:
  - Represent all knowledge you need to reason with
- Inferential adequacy:
  - Allow new knowledge to be inferred from a basic set of facts
    - Don’t innumerate negatives that could be covered by one positive
- Inferential Efficiency:
  - Inferences should be made easily/efficiently

Knowledge Representation Requirements

- Representational adequacy:
  - Represent all knowledge you need to reason with
- Inferential adequacy:
  - Allow new knowledge to be inferred from a basic set of facts
    - Don’t innumerate negatives that could be covered by one positive
- Inferential Efficiency:
  - Inferences should be made easily/efficiently
- Clear Syntax and Semantics:
  - Know allowable expressions of the language and what they mean

Knowledge Representation Requirements

- Representational adequacy:
  - Represent all knowledge you need to reason with
- Inferential adequacy:
  - Allow new knowledge to be inferred from a basic set of facts
    - Don't innumerate negatives that could be covered by one positive
- Inferential Efficiency:
  - Inferences should be made easily/efficiently
- Clear Syntax and Semantics:
  - Know allowable expressions of the language and what they mean
- Naturalness:
  - Reasonably natural and easy to use

Syntax vs. Semantics

- Syntax: The form, grammar, or structure of expressions, statements and program units (i.e. allowable expressions)

Syntax vs. Semantics

- Syntax: The form, grammar, or structure of expressions, statements and program units (i.e. allowable expressions)

- Semantics: The meaning of the expressions, statements and program units

Syntax vs. Semantics

- Syntax: The form, grammar, or structure of expressions, statements and program units (i.e. allowable expressions)

- Semantics: The meaning of the expressions, statements and program units

- Syntax example:
  - blue(Ryan-car) is OK, but Ryan-car(blue & black) is not OK

Syntax vs. Semantics

- Syntax: The form, grammar, or structure of expressions, statements and program units (i.e. allowable expressions)

- Semantics: The meaning of the expressions, statements and program units

- Syntax example:
  - blue(Ryan-car) is OK, but Ryan-car(blue & black) is not OK

- Semantics example:
  - blue(Ryan-car) means, “Ryan’s car is blue”
  - But does not mean (an instruction to), “Paint Ryan’s car blue”
  - Very important for drawing new conclusions

Natural Representation

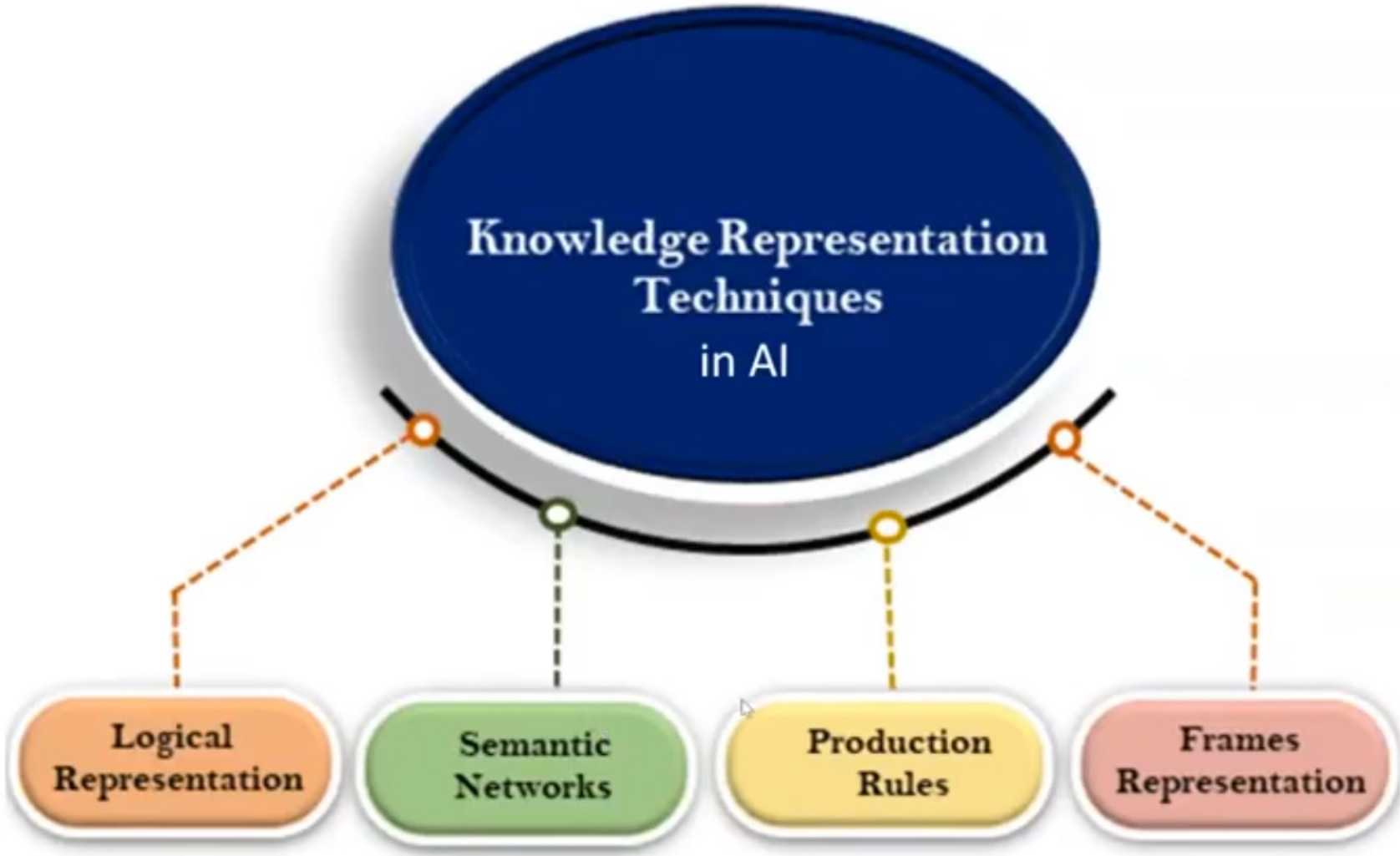
- Choose names for symbols that are meaningful

- Natural Fact:
  - “If someone has a headache they should take aspirin”

Natural Representation

- Choose names for symbols that are meaningful
- Natural Fact:
  - "If someone has a headache they should take aspirin"
- Representation 1: if(x,h,a)
- Representation 2: IF symptom(X, headache) THEN medication(X, aspirin)
- Latter is more readable and easier to deal with

Major Representation Approaches



Basics of Logic

Introducing Logic

- Goals of Logic:
  - Represent knowledge of the world
  - Reason with that knowledge

- Logic is arguably the most important approach to knowledge representation

Introducing Logic

- Goals of Logic:
  - Represent knowledge of the world
  - Reason with that knowledge
- Logic is arguably the most important approach to knowledge representation
- A logic (almost by definition):
  - Has a well-defined syntax and semantics
  - Concerned with truth preserving inference

Introducing Logic

- Goals of Logic:
  - Represent knowledge of the world
  - Reason with that knowledge
- Logic is arguably the most important approach to knowledge representation
- A logic (almost by definition):
  - Has a well-defined syntax and semantics
  - Concerned with truth preserving inference
- Logic studies ways in which humans reason formally through argument
- 'Logic' usually refers to either propositional logic or first order predicate logic

Logic: Argument

- Logical argument:
  - A (finite) set of statements consisting of premises (i.e. facts, knowledge, supporting evidence), and a conclusion

Logic: Argument

- Logical argument:
  - A (finite) set of statements consisting of premises (i.e. facts, knowledge, supporting evidence), and a conclusion
  - Puts reasoning into words
  - Conclusion is identified by ‘therefore’ symbol (∴):

Logic: Argument

- Logical argument:
  - A (finite) set of statements consisting of premises (i.e. facts, knowledge, supporting evidence), and a conclusion
  - Puts reasoning into words
  - Conclusion is identified by 'therefore' symbol (∴):
- Example:

If you overslept, you'll be late.
You aren't late.
∴ You didn't oversleep.

Logic: Argument

- Logical argument:
  - A (finite) set of statements consisting of premises (i.e. facts, knowledge, supporting evidence), and a conclusion
  - Puts reasoning into words
  - Conclusion is identified by 'therefore' symbol (∴):
- Example:

If you overslept, you'll be late.
You aren't late.
∴ You didn't oversleep.

- How to describe the quality of an argument?

Valid Arguments

- Valid Argument:
  - Impossible to have the premises all true and conclusion false

If you overslept, you'll be late.
You aren't late.
∴ You didn't oversleep.

Valid

Valid Arguments

- Valid Argument:
  - Impossible to have the premises all true and conclusion false

If you overslept, you'll be late.
You aren't late.
∴ You didn't oversleep.

Valid

If you overslept, you'll be late.
You didn't oversleep.
∴ You aren't late.

Invalid

Valid Arguments

- Valid Argument:
  - Impossible to have the premises all true and conclusion false

If you overslept, you'll be late.
You aren't late.
∴ You didn't oversleep.

Valid

If you overslept, you'll be late.
You didn't oversleep.
∴ You aren't late.

Invalid

If you're in France, you're in Europe.
You aren't in Europe.
∴ You aren't in France.

Valid

Valid Arguments

- Valid Argument:
  - Impossible to have the premises all true and conclusion false

If you overslept, you'll be late.
You aren't late.
∴ You didn't oversleep.

Valid

If you overslept, you'll be late.
You didn't oversleep.
∴ You aren't late.

Invalid

If you're in France, you're in Europe.
You aren't in Europe.
∴ You aren't in France.

Valid

If you're in France, you're in Europe.
You aren't in France.
∴ You aren't in Europe.

Invalid

Sound Arguments

- Sound Argument:
  - Is valid and every premise is true
- Unsound (2 ways):
  1. Has a false premise
  2. Conclusion might not follow from premises

If you're reading this, you aren't illiterate.
You're reading this.
∴ You aren't illiterate.

Valid and Sound

Sound Arguments

- Sound Argument:
  - Is valid and every premise is true
- Unsound (2 ways):
  1. Has a false premise
  2. Conclusion might not follow from premises

If you're reading this, you aren't illiterate.
You're reading this.
∴ You aren't illiterate.

Valid and Sound

All logicians are millionaires.
Urbanowicz is a logician.
∴ Urbanowicz is a millionaire.

Valid but Unsound

Sound Arguments

- Sound Argument:
  - Is valid and every premise is true
- Unsound (2 ways):
  1. Has a false premise
  2. Conclusion might not follow from premises

If you're reading this, you aren't illiterate.
You're reading this.
∴ You aren't illiterate.

Valid and Sound

All logicians are millionaires.
Urbanowicz is a logician.
∴ Urbanowicz is a millionaire.

Valid but Unsound

All millionaires eat well.
Urbanowicz eats well.
∴ Urbanowicz is a millionaire.

Invalid thus Unsound

Syllogistic Logic

- Invented by Aristotle
- Syllogistic language:
  - Capital letters for general categories
  - Lower case for specific items/individuals

Syllogistic Logic

- Invented by Aristotle
- Syllogistic language:
  - Capital letters for general categories
  - Lower case for specific items/individuals

All logicians are smart.
Urbanowicz is a logician.
∴ Urbanowicz is smart.

Syllogistic Logic

- Invented by Aristotle
- Syllogistic language:
  - Capital letters for general categories
  - Lower case for specific items/individuals

All logicians are smart.	all L is S
Urbanowicz is a logician.	u is L
∴ Urbanowicz is smart.	∴ u is S

Syllogistic Logic

- Invented by Aristotle
- Syllogistic language:
  - Capital letters for general categories
  - Lower case for specific items/individuals

All logicians are smart.
Urbanowicz is a logician.
∴ Urbanowicz is smart.

all L is S
u is L
∴ u is S

- Well-formed formulas (*wffs*):
  - Syntax (acceptable objects) of logic
  - Use five words: all, no, some, is, not
  - *Wffs* have any of these 8 forms:

Syllogistic Logic

- Invented by Aristotle
- Syllogistic language:
  - Capital letters for general categories
  - Lower case for specific items/individuals

All logicians are smart. Urbanowicz is a logician. ∴ Urbanowicz is smart.	
all L is S u is L ∴ u is S	

- Well-formed formulas (*wffs*):
  - Syntax (acceptable objects) of logic
  - Use five words: all, no, some, is, not
  - *Wffs* have any of these 8 forms:

all A is B	some A is B	x is A	x is y
no A is B	some A is not B	x is not A	x is not y

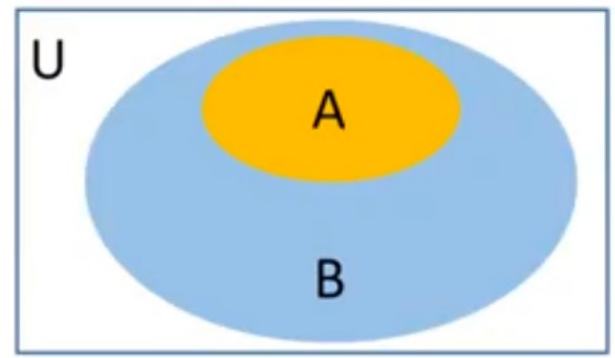
Wffs with Venn Diagrams

- **Venn Diagram**: A diagram consisting of various overlapping figures contained in a rectangle called the universe (U)

Wffs with Venn Diagrams

- **Venn Diagram**: A diagram consisting of various overlapping figures contained in a rectangle called the universe (U)

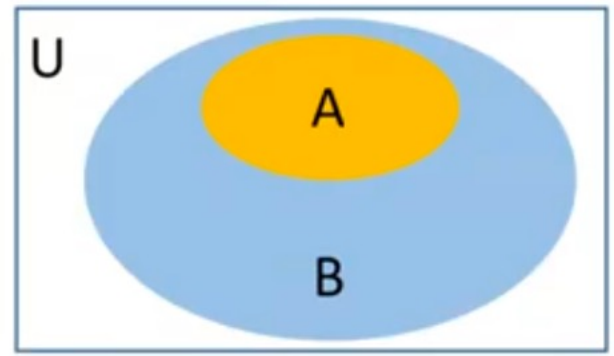
- **all A is B**. (If A, then B)



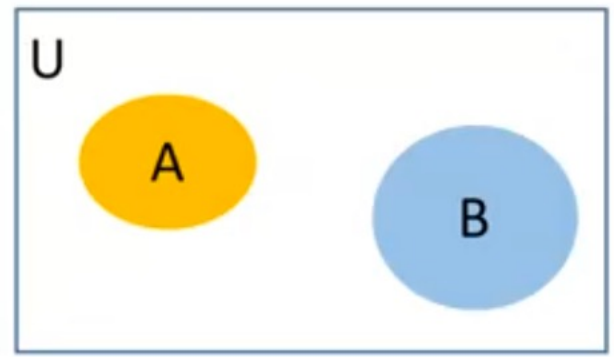
Wffs with Venn Diagrams

- **Venn Diagram**: A diagram consisting of various overlapping figures contained in a rectangle called the universe (U)

- **all A is B**. (If A, then B)



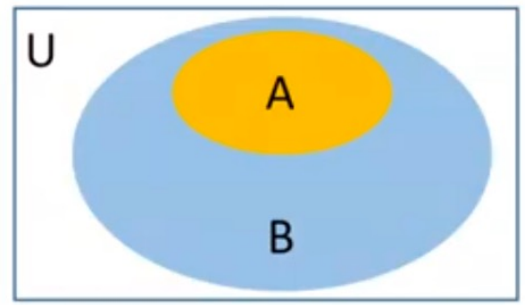
- **No A is B**



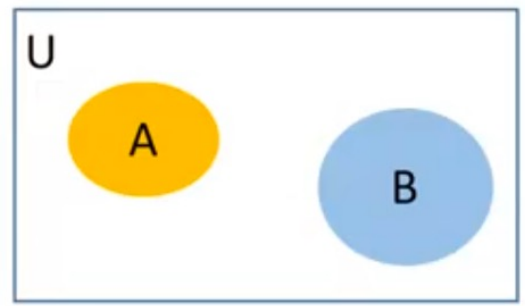
Wffs with Venn Diagrams

- **Venn Diagram**: A diagram consisting of various overlapping figures contained in a rectangle called the universe (U)

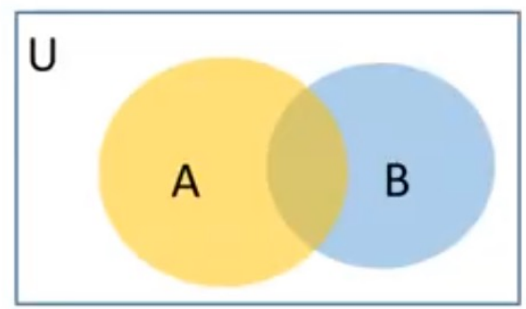
- **all A is B**. (If A, then B)



- **No A is B**



- **Some A is B**



Syllogism

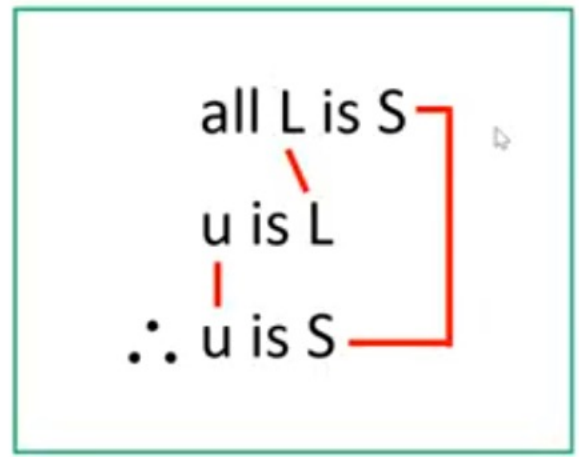
- Arguments of two or more *wffs* in which each letter occurs twice and the letters "form a chain"

Syllogism

- Arguments of two or more *wffs* in which each letter occurs twice and the letters "form a chain"
    - Each *wff* has at least one letter in common with the *wff* just below it, and first *wff* has at least one letter in common with the last *wff*

Syllogism

- Arguments of two or more *wffs* in which each letter occurs twice and the letters "form a chain"
  - Each *wff* has at least one letter in common with the *wff* just below it, and first *wff* has at least one letter in common with the last *wff*



Major premise

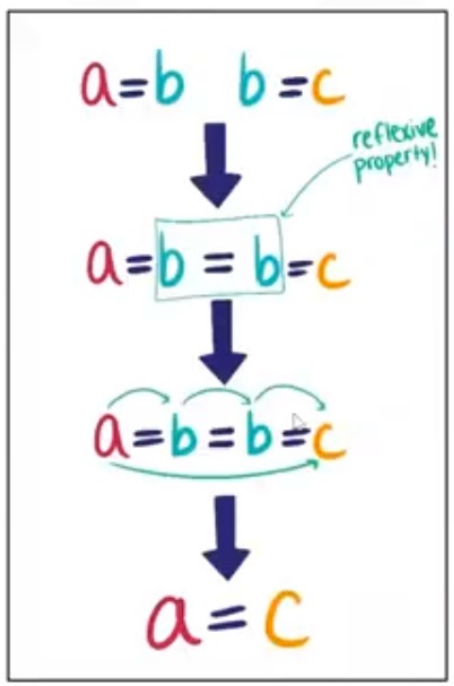
Minor premise

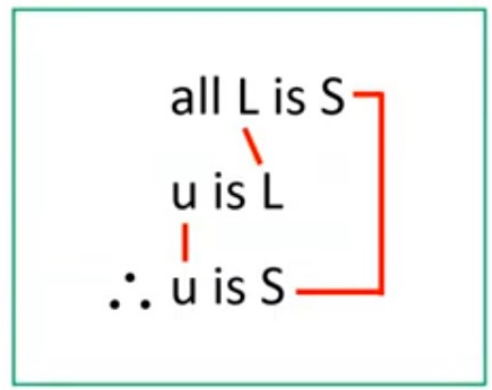
Conclusion

Syllogism

- Arguments of two or more *wffs* in which each letter occurs twice and the letters "form a chain"
  - Each *wff* has at least one letter in common with the *wff* just below it, and first *wff* has at least one letter in common with the last *wff*

Transitive Property





Major premise

Minor premise

Conclusion

The Star Test

- Gimmick testing a syllogism for validity

The Star Test

- Gimmick testing a syllogism for validity
- Identify ‘distributed’ letters:
  - Occurs just after ‘all’ or anywhere after ‘no’ or ‘not’
  - The first letter after ‘all’, but not the second
  - Both letters after ‘no’
  - Any letter after ‘not’

all A is B	some A is B	x is A	x is y
no A is B	some A is not B	x is not A	x is not y

The Star Test

- Gimmick testing a syllogism for validity
- Identify ‘distributed’ letters:
  - Occurs just after ‘all’ or anywhere after ‘no’ or ‘not’
  - The first letter after ‘all’, but not the second
  - Both letters after ‘no’
  - Any letter after ‘not’

all A is B	some A is B	x is A	x is y
no A is B	some A is not B	x is not A	x is not y

- Star Test:
  - ‘*’ premise letters that are distributed and conclusion letters that aren’t

The Star Test

- Gimmick testing a syllogism for validity
- Identify 'distributed' letters:
  - Occurs just after 'all' or anywhere after 'no' or 'not'
  - The first letter after 'all', but not the second
  - Both letters after 'no'
  - Any letter after 'not'

all A is B	some A is B	x is A	x is y
no A is B	some A is not B	x is not A	x is not y

- Star Test:
  - '*' premise letters that are distributed and conclusion letters that aren't
  - Syllogism is valid if:
    - Every capital letter is starred exactly once
    - There is exactly one star on the right hand side

all L* is S

u is L

∴ u is S*

Valid

The Star Test

- Gimmick testing a syllogism for validity
- Identify 'distributed' letters:
  - Occurs just after 'all' or anywhere after 'no' or 'not'
  - The first letter after 'all', but not the second
  - Both letters after 'no'
  - Any letter after 'not'

all A is B	some A is B	x is A	x is y
no A is B	some A is not B	x is not A	x is not y

- Star Test:
  - '*' premise letters that are distributed and conclusion letters that aren't
  - Syllogism is valid if:
    - Every capital letter is starred exactly once
    - There is exactly one star on the right hand side

all L* is S
u is L
∴ u is S*

Valid

no A* is B*
no C* is A*
∴ no C is B

Invalid

The Star Test

- Gimmick testing a syllogism for validity
- Identify 'distributed' letters:
  - Occurs just after 'all' or anywhere after 'no' or 'not'
  - The first letter after 'all', but not the second
  - Both letters after 'no'
  - Any letter after 'not'

all A is B	some A is B	x is A	x is y
no A is B	some A is not B	x is not A	x is not y

- Star Test:
  - '*' premise letters that are distributed and conclusion letters that aren't
  - Syllogism is valid if:
    - Every capital letter is starred exactly once
    - There is exactly one star on the right hand side

all L* is S	no A* is B*	no P* is B*
u is L	no C* is A*	some C is B
∴ u is S*	∴ no C is B	∴ some C* is not P
Valid	**Invalid**	**Valid**

Enthymeme

- Argument in which the major premise is left unstated
  - Often a conclusion supported by a single premise

Enthymeme

- Argument in which the major premise is left unstated
  - Often a conclusion supported by a single premise

- Examples:

  She must be a good student since she is on the Dean's list

  [Conclusion] [Minor Premise]

  She must be a good student since she is on the Dean's list

Enthymeme

- Argument in which the major premise is left unstated
  - Often a conclusion supported by a single premise

- Examples:

  She must be a good student since she is on the Dean's list

[Conclusion] [Minor Premise]

She must be a good student since she is on the Dean's list

[Implied Major Premise]

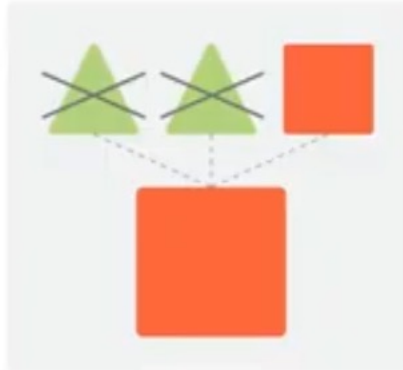
All good students are on the Dean's list

Overview of Reasoning

Types of Reasoning

Deductive	Inductive	Abductive
All men are mortal;	Socrates is a man;	All men are mortal;
Socrates is a man;	Socrates is mortal;	Socrates is mortal;
∴ Socrates is mortal.	∴ All men are mortal.	∴ Socrates is a man.

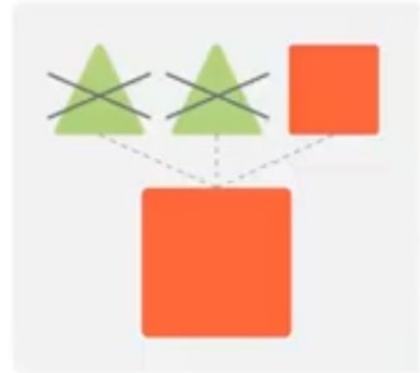
Deductive Reasoning



- General → Specific i.e. “Top-down logic”

- Reasoning from one or more statements to reach a logically certain conclusion

Deductive Reasoning

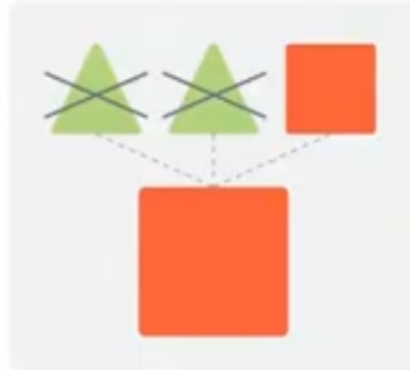


- General → Specific i.e. “Top-down logic”

- Reasoning from one or more statements to reach a logically certain conclusion

- Reasoning goes in the same direction as that of the conditionals
  - Opposite is true for abductive reasoning

Deductive Reasoning



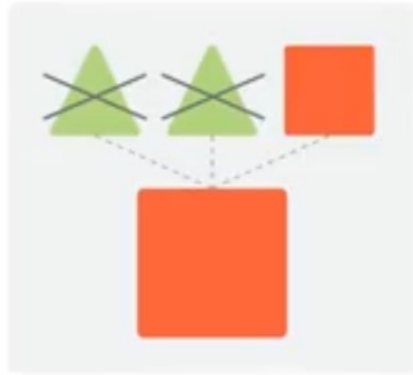
- General → Specific i.e. “Top-down logic”

- Reasoning from one or more statements to reach a logically certain conclusion

- Reasoning goes in the same direction as that of the conditionals
  - Opposite is true for abductive reasoning

- A conclusion is reached reductively:
  - Applying general rules which hold over the entirety of a closed domain, narrowing the range under consideration until only the conclusion(s) is left

Deductive Reasoning



- General → Specific i.e. “Top-down logic”

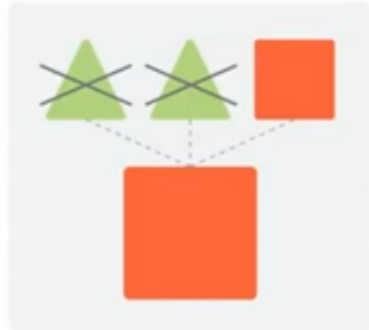
- Reasoning from one or more statements to reach a logically certain conclusion

- Reasoning goes in the same direction as that of the conditionals
  - Opposite is true for abductive reasoning

- A conclusion is reached reductively:
  - Applying general rules which hold over the entirety of a closed domain, narrowing the range under consideration until only the conclusion(s) is left

- Syllogisms are a common form of deductive reasoning

Deductive Reasoning



- General → Specific i.e. “Top-down logic”

- Reasoning from one or more statements to reach a logically certain conclusion

- Reasoning goes in the same direction as that of the conditionals
  - Opposite is true for abductive reasoning

- A conclusion is reached reductively:
  - Applying general rules which hold over the entirety of a closed domain, narrowing the range under consideration until only the conclusion(s) is left

- Syllogisms are a common form of deductive reasoning

- Hallmark of expert systems

Inductive Reasoning

- Specific → General i.e. “Bottom-up logic”

- Reasons from evidence about *some* members of a class in order to form a conclusion about *all* members of a class

Inductive Reasoning

- Specific → General i.e. “Bottom-up logic”

- Reasons from evidence about *some* members of a class in order to form a conclusion about *all* members of a class

- A conclusion derived through inductive reasoning is called a hypothesis
  - Always less certain than evidence itself
  - i.e. conclusion is probable but uncertain

Inductive Reasoning

- Specific → General i.e. “Bottom-up logic”

- Reasons from evidence about *some* members of a class in order to form a conclusion about *all* members of a class

- A conclusion derived through inductive reasoning is called a hypothesis
  - Always less certain than evidence itself
  - i.e. conclusion is probable but uncertain

- Inductive reasoning here is not the same as induction in mathematical proofs
  - Mathematical induction is actually a form of deductive reasoning

Inductive Reasoning

- Specific → General i.e. “Bottom-up logic”

- Reasons from evidence about *some* members of a class in order to form a conclusion about *all* members of a class

- A conclusion derived through inductive reasoning is called a hypothesis
  - Always less certain than evidence itself
  - i.e. conclusion is probable but uncertain

- Inductive reasoning here is not the same as induction in mathematical proofs
  - Mathematical induction is actually a form of deductive reasoning

- Machine learning is all about inductive reasoning
  - Using a set of specific instances to find useful generalizations

Inductive Reasoning

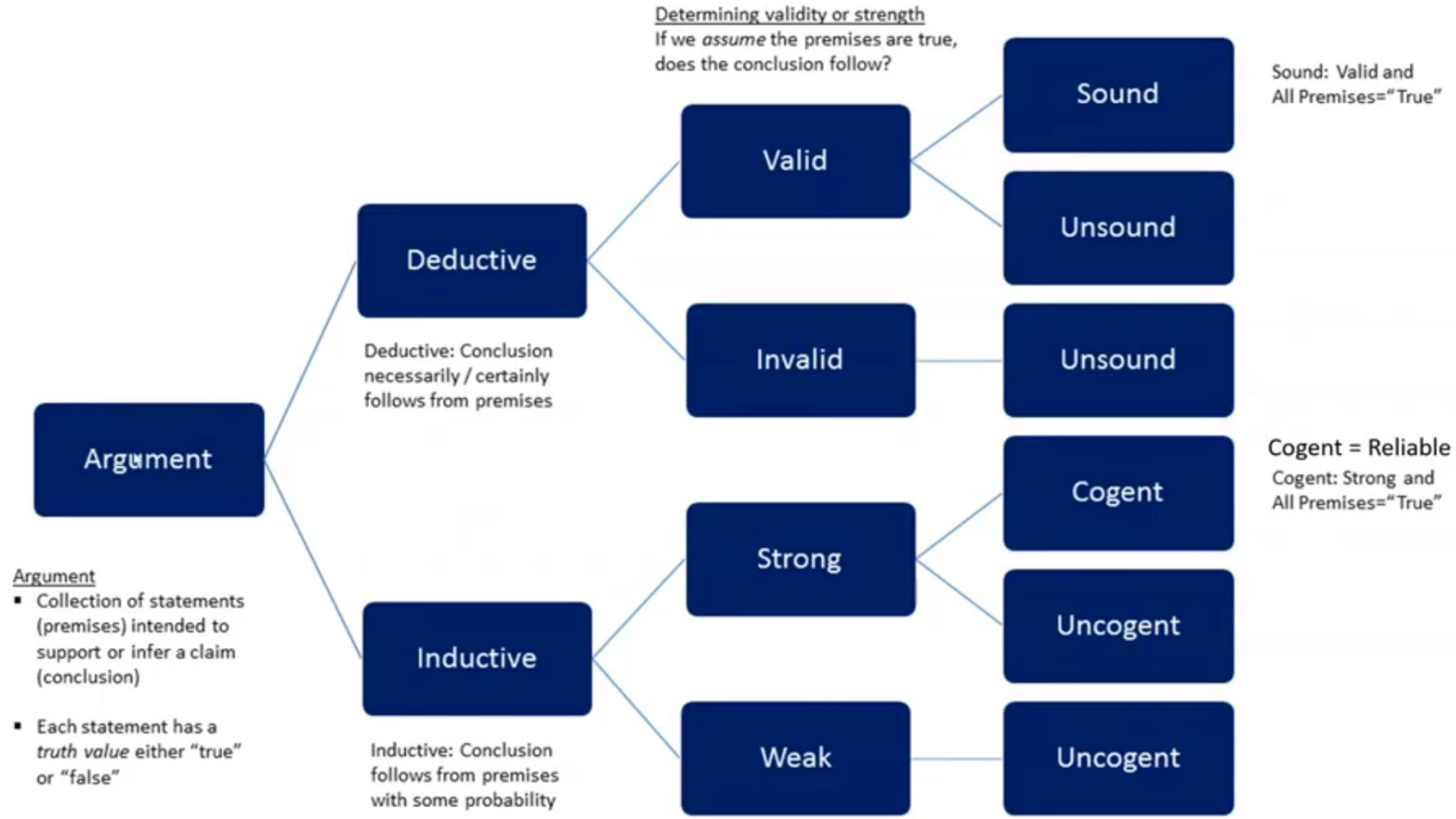
- Statistical Syllogism:
  - Correct premise/conclusion link: Strong vs weak
  - True premises: Cogent/reliable vs uncogent/unreliable

Inductive Reasoning

- Statistical Syllogism:
  - Correct premise/conclusion link: Strong vs weak
  - True premises: Cogent/reliable vs uncogent/unreliable

90 percent of AT shelters have water.
Rocky Gap is an AT shelter.
That's all we know about the matter.
∴ It's 90 percent probable that Rocky Gap has water

Quality of Reasoning Arguments



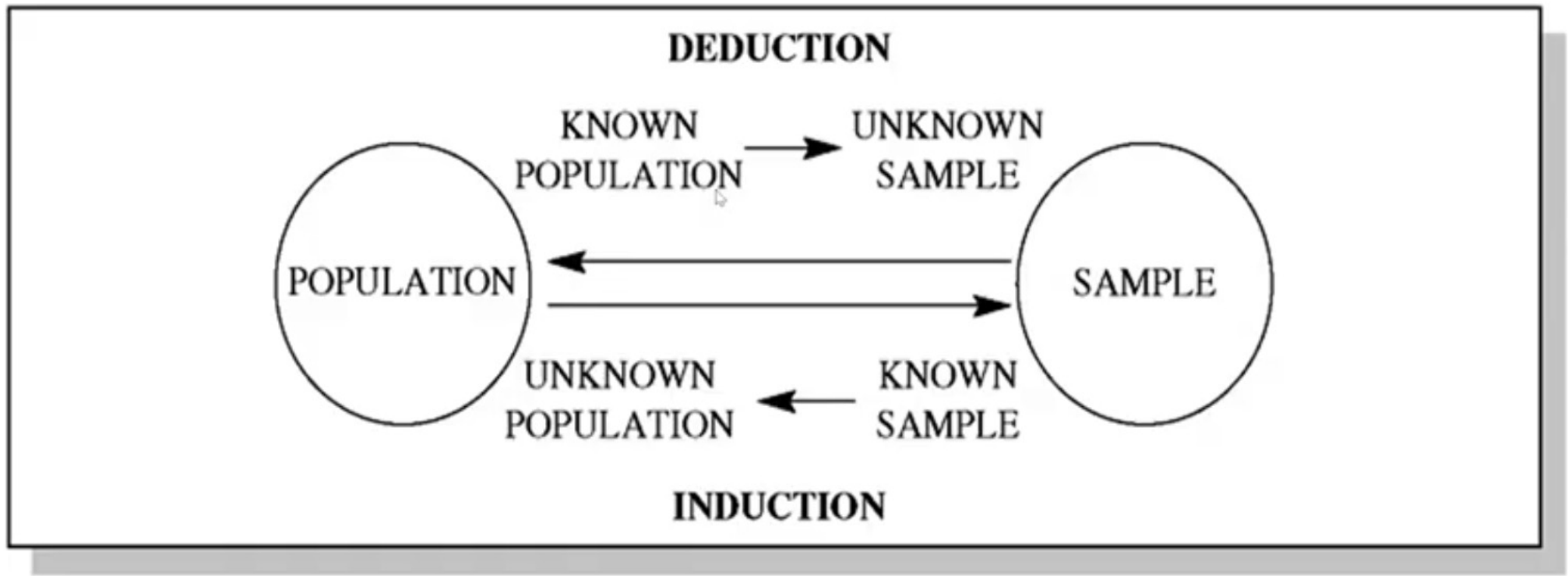
Argument

- Collection of statements (premises) intended to support or infer a claim (conclusion)
- Each statement has a *truth value* either "true" or "false"

Source Information: Patrick J. Hurley "A Concise Introduction to Logic-12th Ed."

Deduction vs Induction

- In context of populations and samples



Deductive vs Inductive Learning

- Deductive learning/instruction:
  - Teacher presents generalization/rule
  - Learner applies this to specific examples or activities

- Inductive learning/instruction:
  - Learners detecting or noticing patterns and working out a ‘rule’ for themselves

Deductive

Generalization (or Rule) ----------> Specific Examples or Activities

Inductive

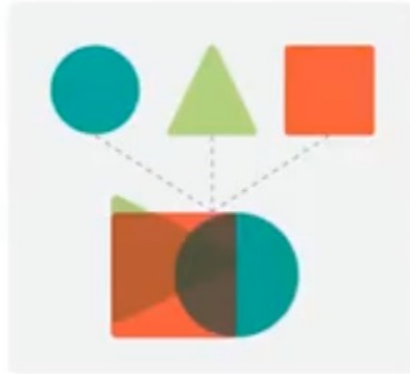
Specific Examples or Activities ----------> Generalization (or Rule)

Abductive Reasoning

- “Inference to the best explanation”

- Starts with an observation or set of observations then seeks to find the simplest and most likely explanation for the observations

Abductive Reasoning



- “Inference to the best explanation”

- Starts with an observation or set of observations then seeks to find the simplest and most likely explanation for the observations

- Yields a plausible conclusion but does not positively verify it - "best available" or "most likely."

Abductive Reasoning

- “Inference to the best explanation”
- Starts with an observation or set of observations then seeks to find the simplest and most likely explanation for the observations
- Yields a plausible conclusion but does not positively verify it - "best available" or "most likely."
- Sometimes used by diagnostic expert systems

Learning Python Coding

Python Coding: Editors and IDEs

- Editor (i.e. text editor):
  - Simply a place where text and code can be written and edited.

- Integrated development environment (IDE):
  - A software application that provides comprehensive facilities to computer programmers for software development.
    - Source code editor
    - Build automation tools
    - Debugger.

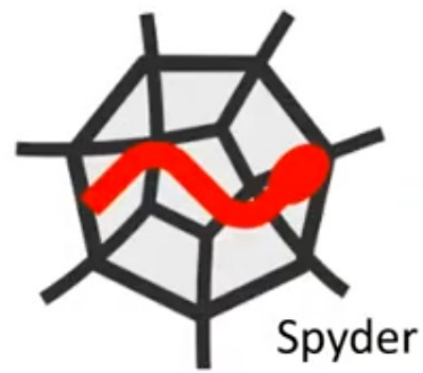
Python Coding: Editors and IDEs

- Editor (i.e. text editor):
  - Simply a place where text and code can be written and edited.

- Integrated development environment (IDE):
  - A software application that provides comprehensive facilities to computer programmers for software development.
    - Source code editor
    - Build automation tools
    - Debugger.

- General Editors and IDEs with Python Support
  - Eclipse + PyDev
  - Sublime Text
  - Atom
  - GNU Emacs
  - Vi / Vim
  - Visual Studio
  - Visual Studio Code
  - Jupyter Notebook

- Python specific IDEs:
  - PyCharm
  - Spyder
  - Thonny
  - IDLE

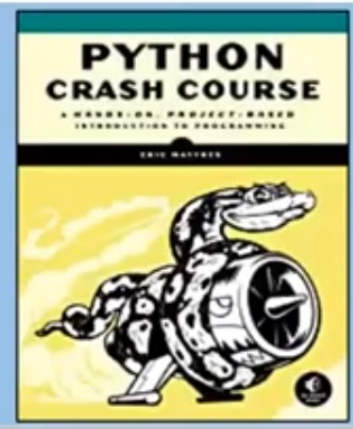


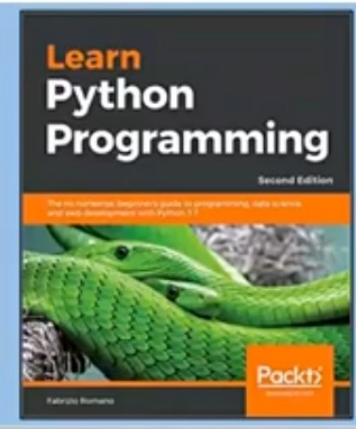
Introductory Tutorials on Python Coding

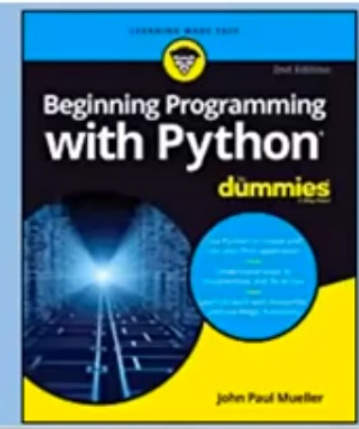
- Self-Guided 'Basics':
  - https://www.learnpython.org/
- Video (Basics):
  - Link: Python Tutorials by Corey Schafer
- Pandas (Data Science):
  - Link: Pandas Tutorial
- Scikit-Learn (Machine Learning):
  - Scikit-learn Tutorial
- Course:
  - BMIN 525 – Introduction to Python Programming (Fall)

Python Books

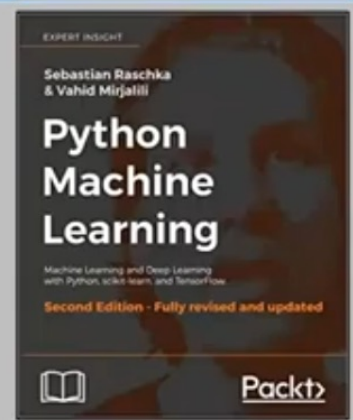
- Python Basics:

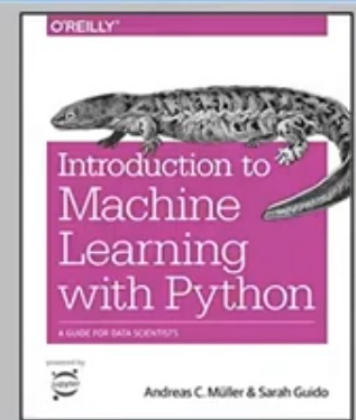




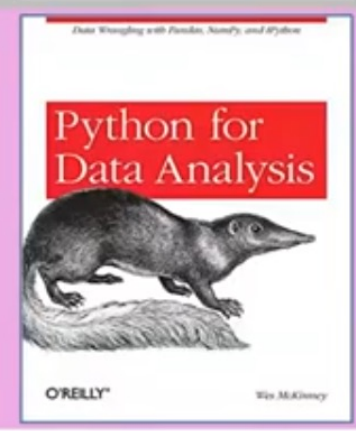


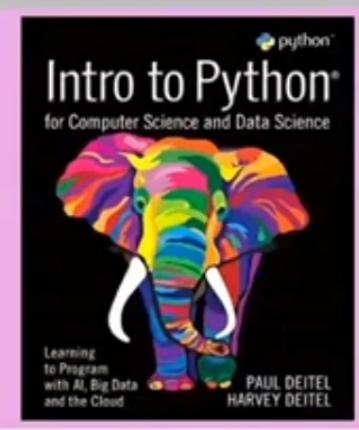
- Python Machine Learning:





- Python Data Science:





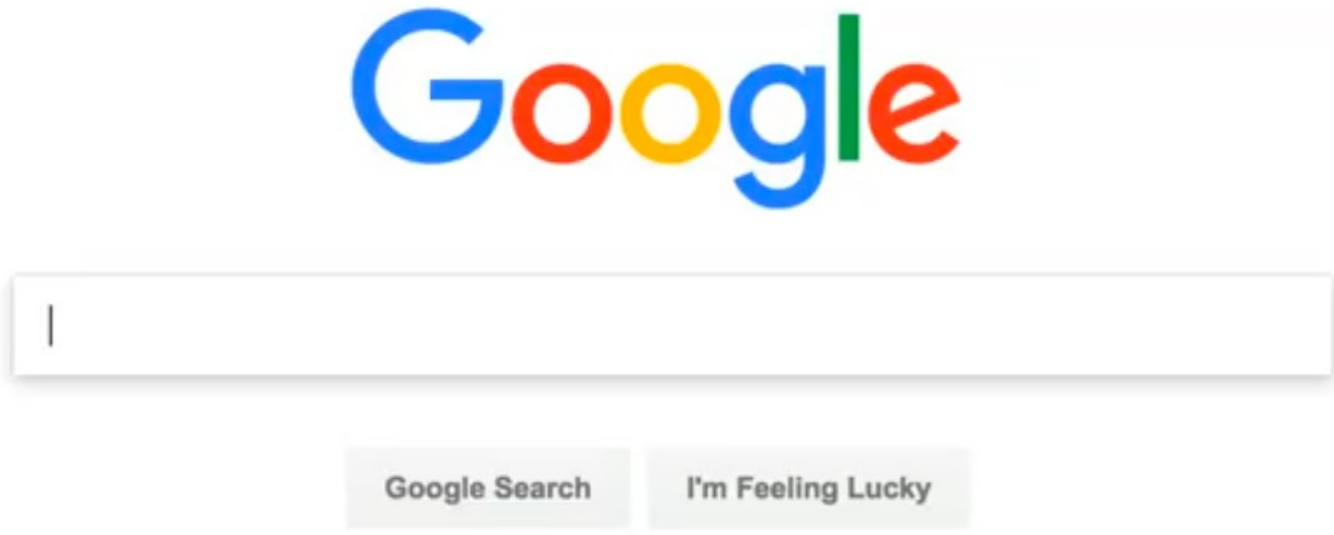
Coding Help

Getting Help with Coding

- Being a data scientist includes being able to be able to seek answers on your own
  - Put some effort into it
  - Don’t be afraid of data/errors
  - Admission of not knowing something is OK

- Most questions you have have already been answered on the web

Where to Get Help



- Hardest part can be figuring out what to search for

Where Else to Get Help Coding/Debugging

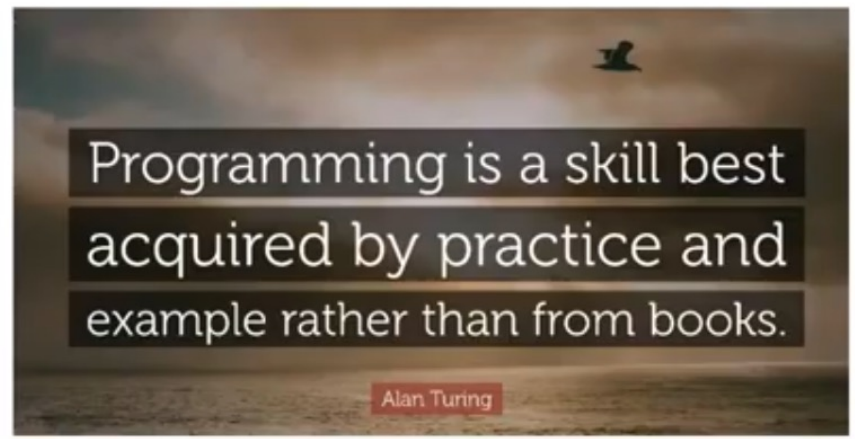
- http://stackoverflow.com

- https://www.geeksforgeeks.org/

GeeksforGeeks
A computer science portal for geeks

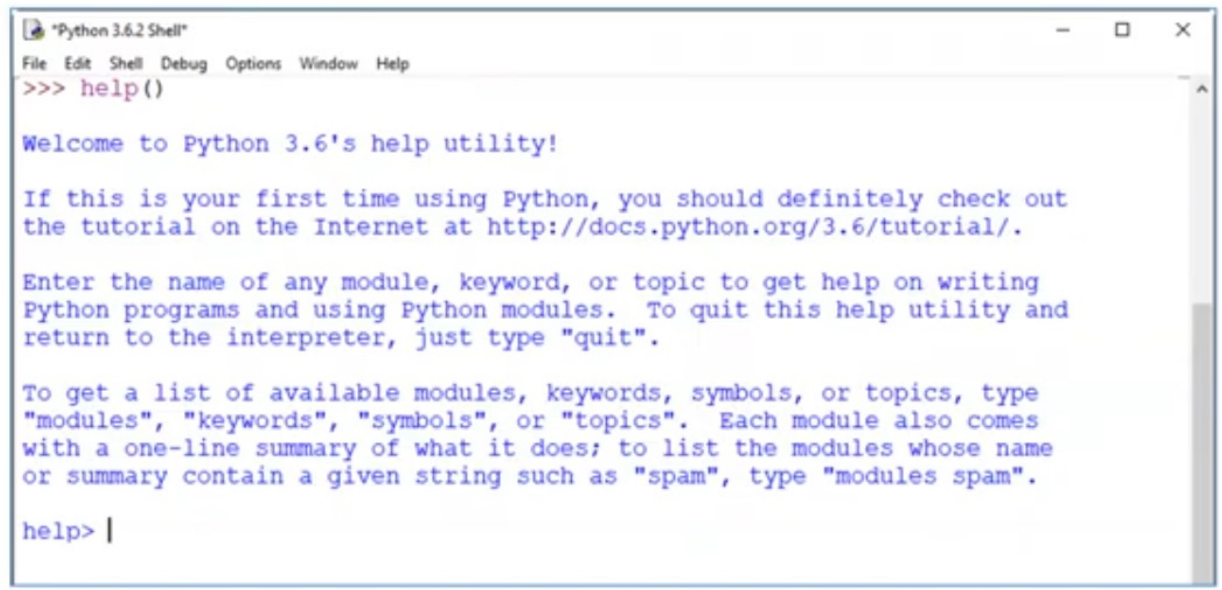
- https://python-forum.io/index.php

- https://www.quora.com/



Describing a Problem

- Give details of question, but be brief!
  - Language (Python)
  - Environment (version, package name, etc)
  - Code you tried
  - Error message
  - Try different ways of explaining what you are trying to do



Git and GitHub

git : Version Control

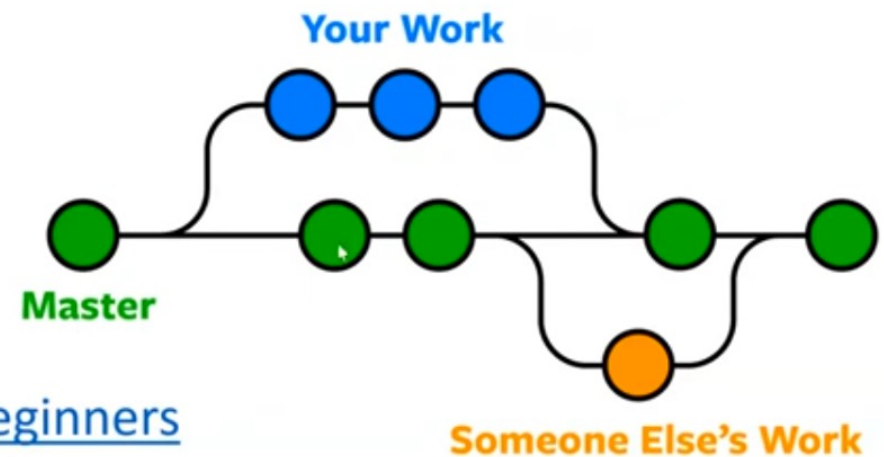
- Version Control:
  - Allows tracking/restoring of files over time
  - Efficient way to keep annotation and track errors
  - Great for projects with multiple contributors

git : Version Control

- Version Control:
  - Allows tracking/restoring of files over time
  - Efficient way to keep annotation and track errors
  - Great for projects with multiple contributors
- Git:
  - Popular open source version control system
  - Command line based
  - Files related to a project are stored on a local computer “repository”

git : Version Control

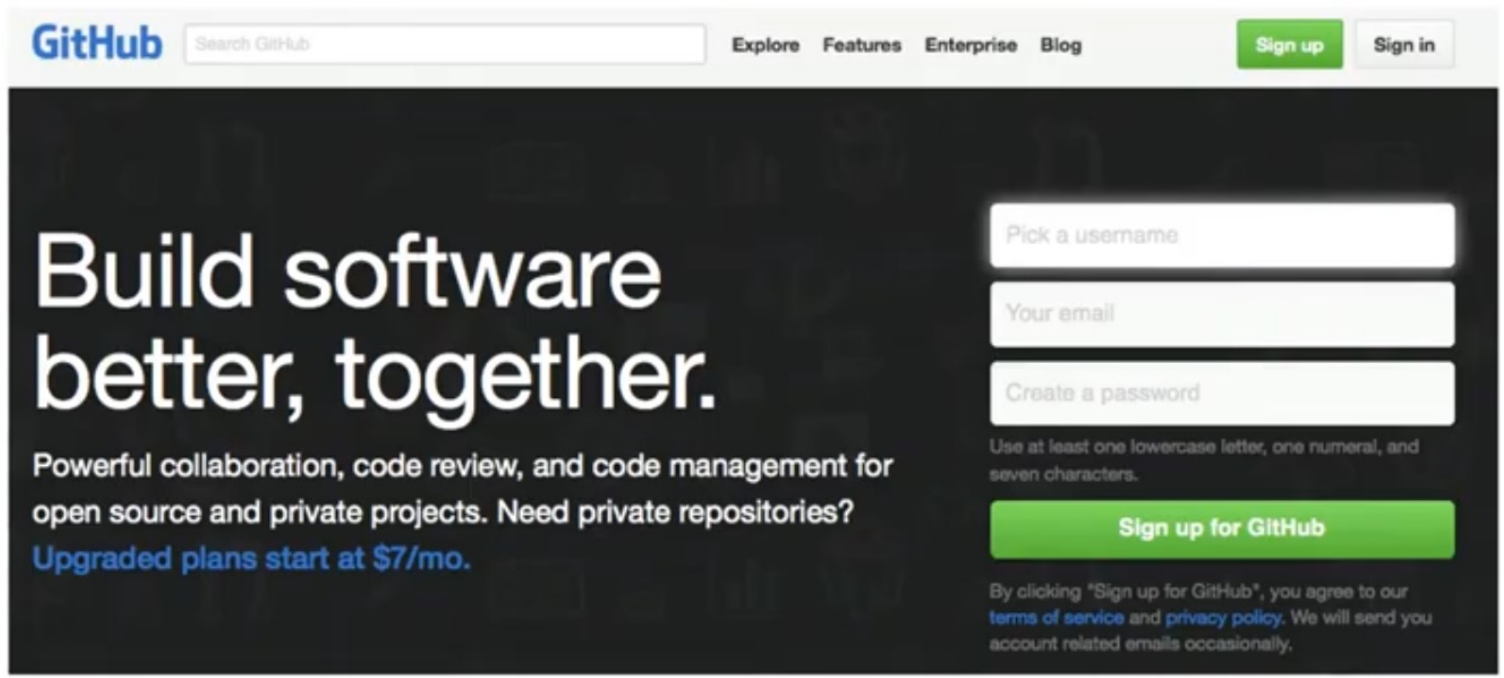
- Version Control:
  - Allows tracking/restoring of files over time
  - Efficient way to keep annotation and track errors
  - Great for projects with multiple contributors
- Git:
  - Popular open source version control system
  - Command line based
  - Files related to a project are stored on a local computer “repository”



Link: An Introduction to Git and GitHub for Beginners

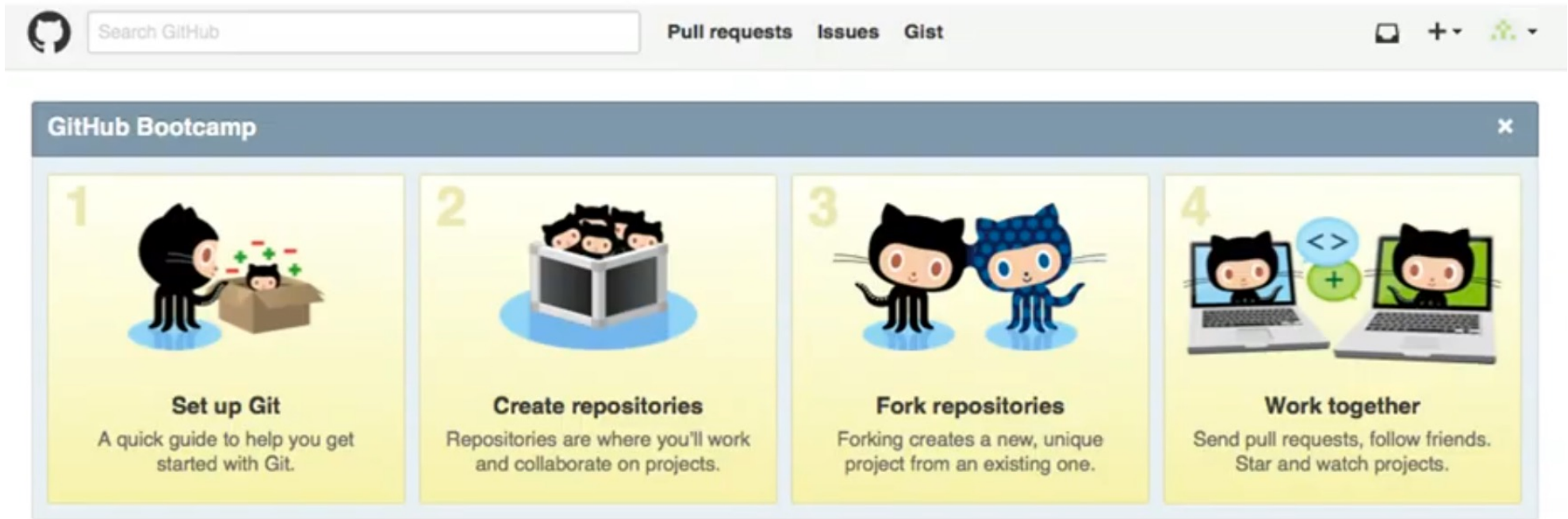
GitHub

- Web-based platform that hosts projects using Git for version control
- Can share code with others and access others' code
- Repositories (i.e. 'repos') are folders which contain intentional snapshots of progress called 'commits

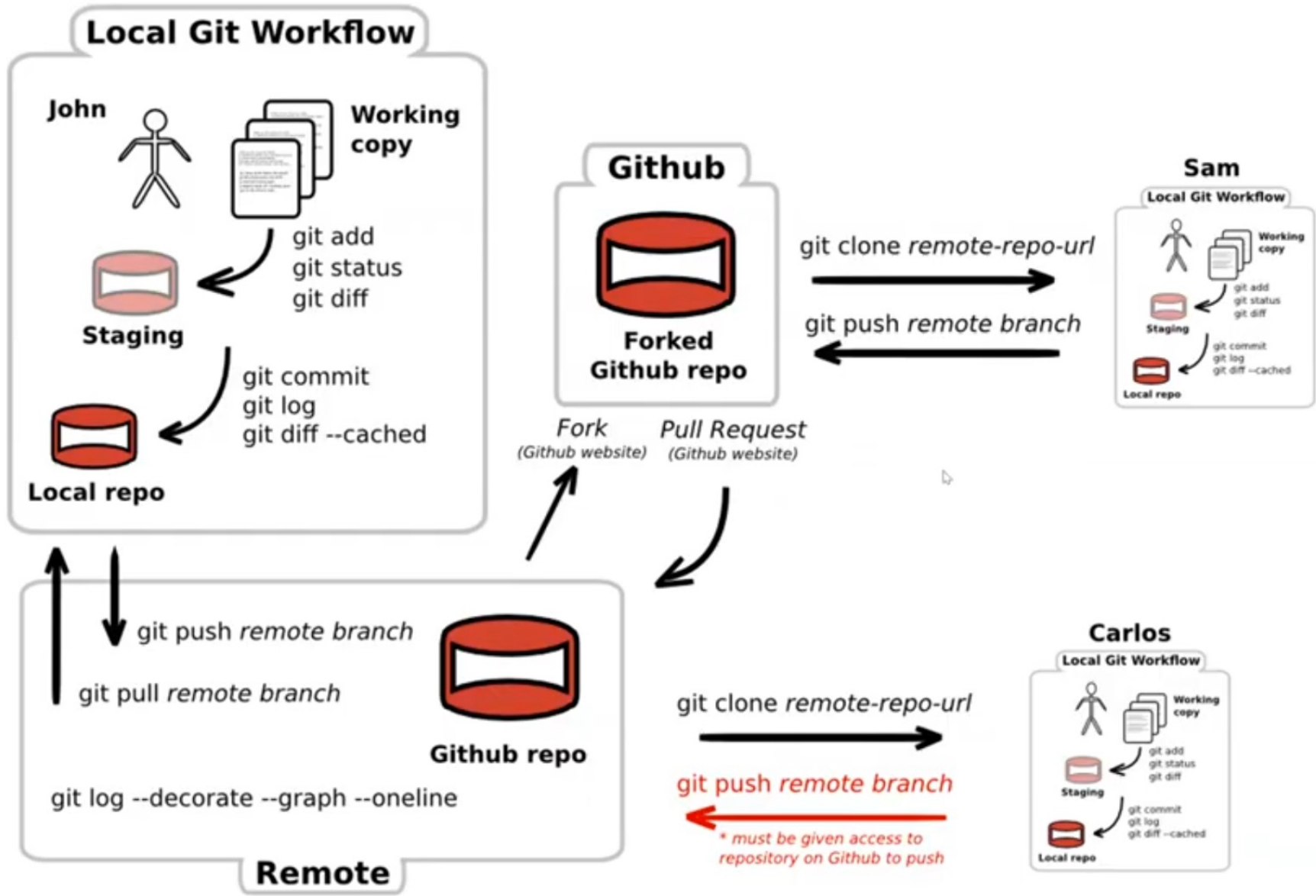


Creating GitHub Repos

- Create:
  - A new one (from scratch)
  - Fork: Copy another user's repository

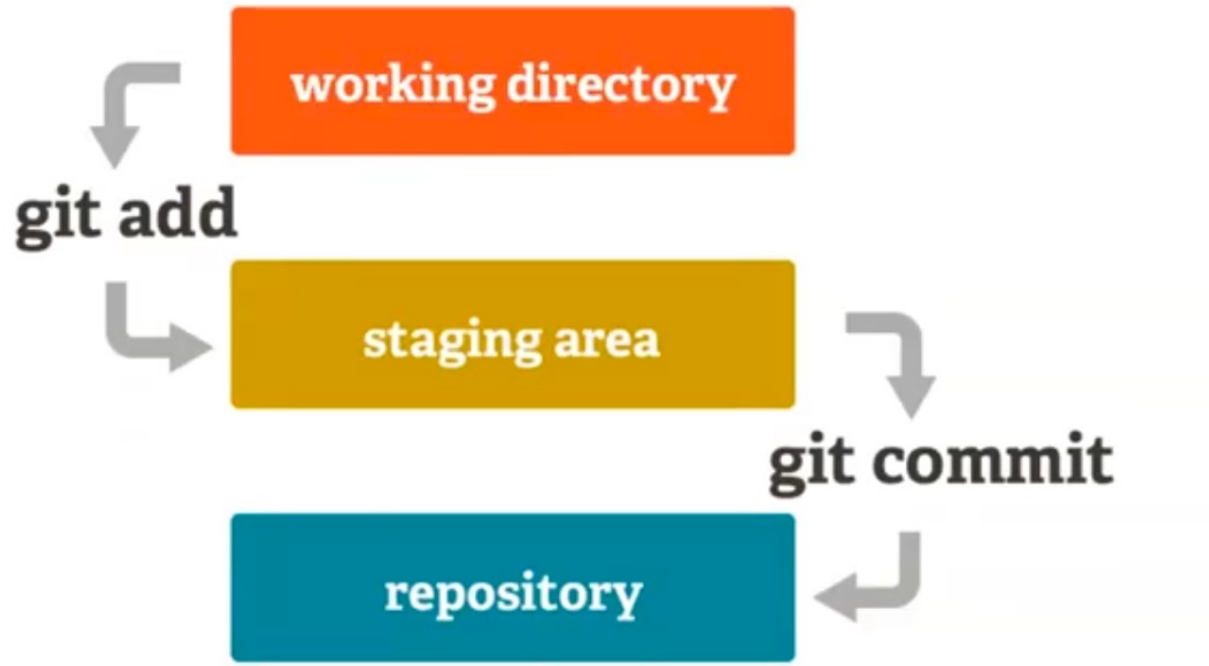


Git/GitHub Workflow Overview



Git Local Commands

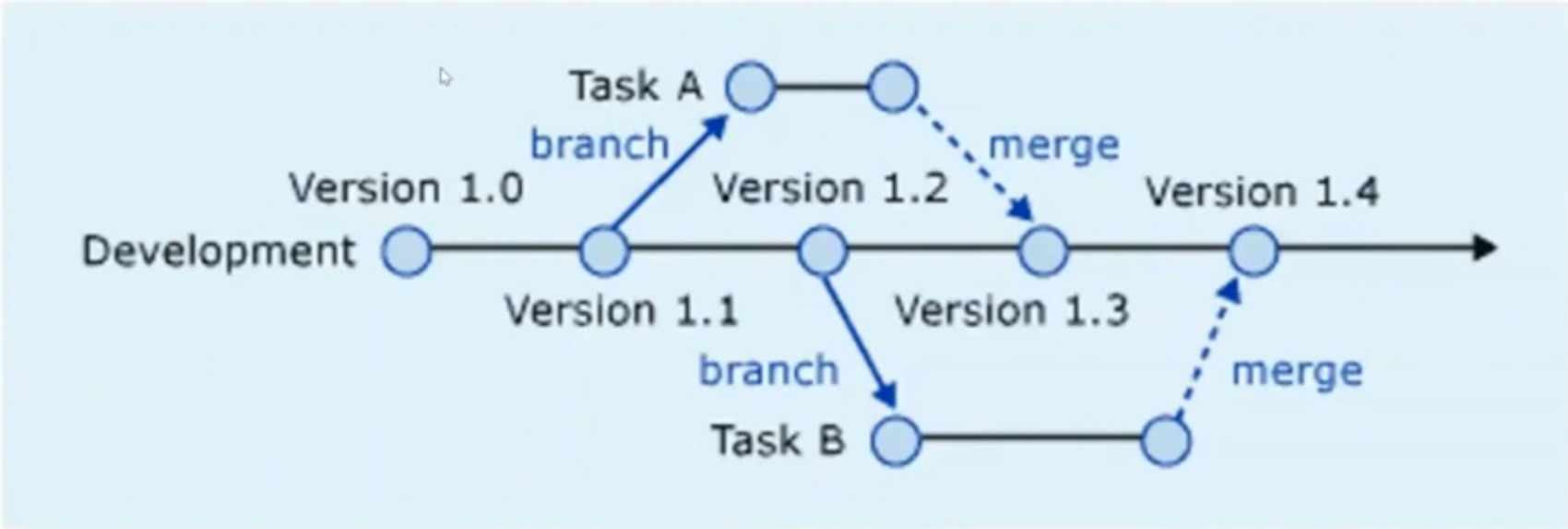
- Making and documenting code updates
  - Add:
    - Select the changes that will be staged for the next commit
  - Commit:
    - Create a snapshot of the staged changes along a timeline of a Git projects history.



Git Version Control Commands

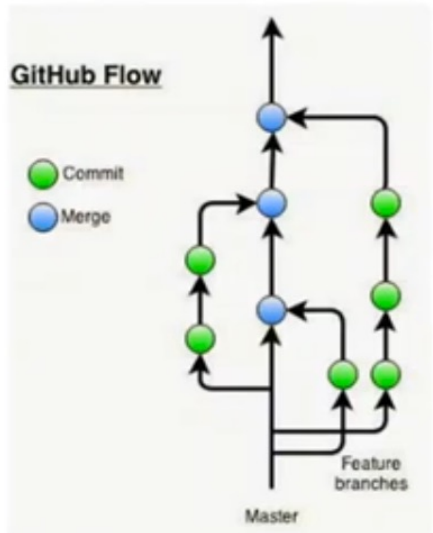
- Branch: create, list, rename, and delete branches

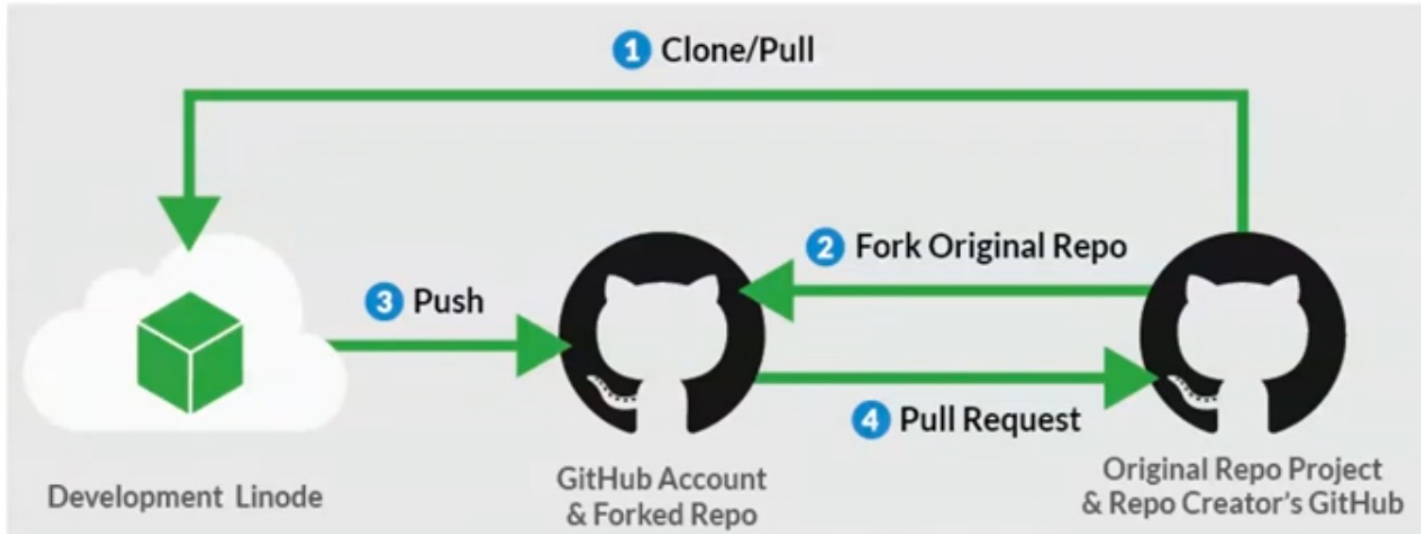
- Merge: Take the independent lines of development created by git branch and integrate them into a single branch



Git Network Commands

- Clone: Target an existing repository and create a clone, or copy of the target repository
- Pull: Update the local version of a repository from a remote
  - Pull request: Asking the owner to pull updated code you provide
- Push: Upload local repository content to a remote repository. Pushing is how you transfer commits from your local repository to a remote repo. It's the counterpart to 'fetch'
- Fetch: Download contents from a remote repository





Assignments

- Reminder: Assignment 1

- Goals:
  - Get your computer set up (and equipped) with the tools needed for using Python and available packages
  - Understand the difference between Anaconda, MiniConda, and Conda
  - Understand how to access Python through your terminal (MacOS) or command line (Windows, Linux)
  - Learn how to use Git and GitHub
  - Learn basics of Python programming