

Instructional Design for Generation AI Lesson Planning



Joe Garner
The MITRE Corporation



Graphics on pp. 7 and 9 by Mickey Stinnett

Contributors: Ali Zaidi and Marilyn Kupetz

Executive Summary

With the Generation AI Nexus initiative (GenAI), MITRE is giving learners—students preparing for the workforce and experienced professionals alike—access to AI tools and the chance to become schooled, creative, world-class AI specialists, data analysts, and machine learning experts.

We are sharing this instructional design guide with instructors—you!—to help you integrate machine learning and data science lessons into your courses as you participate in GenAI. The guide offers a high-level view of instructional design methodology and demonstrates these methods with a practical, real-world example.

After introducing the methodology, we'll step through the example, which integrates an AI-based lesson in machine learning into a computer science course—one that is accessible through the GenAI Nexus and that you can adapt for your own purposes.

We close by presenting simple job aids to help you write outcomes and learning objectives for your lessons and to then share your suggestions in online repositories such as GitHub and the Gen AI Nexus.

We welcome your feedback. Write to us at generationainexus@mitre.org

Table of Contents

Overview	5
Lesson Planning Methodology: Understanding by Design.....	6
Understanding by Design as a Lesson Integration Strategy.....	6
Stage 1: Identify the Results You Want.....	7
Stage 2: Determine Acceptable Evidence by Thinking Like an Assessor	8
Common Assessment Techniques and Activities	8
Stage 3: Plan Learning Experiences and Instruction.....	9
Integrating a Machine Learning or Data Science Lesson into a Computer Science Course.....	10
Review Existing Module and Add Outcome.....	11
Ask Essential Questions for Each Outcome.....	12
Use Essential Questions to Create Your Learning Objectives.....	13
Map Assessments to Your Learning Objectives	14
Add Content and Course Resources.....	15
Delivering Instruction	16
ROPES Model.....	16
Sample Lesson from Project Management Course	17
Checking for Success	18
Sample Data Science Lesson in Simple Cluster Analysis.....	19
Online Resources for Jupyter Notebook, Python, and Machine Learning.....	22
APPENDIX: Writing Impactful Objectives.....	23
Six Facets of Understanding	23
Bloom’s Taxonomy.....	25
References	26

List of Figures

Figure 1. Backward Design (adapted from Wiggins & McTighe, 2005).....	7
Figure 2. UbD Curricular Priorities (adapted from Wiggins & McTighe, 2005)	9
Figure 3. Partial Syllabus for Computational Science Tools Course.....	10
Figure 4. Bloom's Taxonomy	25

Overview

Your data science and machine learning lessons are likely to be more successful if you develop them using instructional design principles such as *establishing outcomes*, *aligning lessons to learning objectives*, *asking questions that lead to enduring understanding*, *prioritizing the learning objectives*, and *mapping assessments to outcomes*.

To this end, here's what you'll find in this guidebook:

- A brief overview of *Understanding by Design (UbD)*, an instructional design methodology for building assessment-based lessons.
- An application of UbD methodology to a real-world example to show you how to decompose a module of instruction, add a lesson, and bring it back into alignment with your outcomes.
- An example applying ROPES model to pace your instruction
- An illustration of a course tool—Jupyter Notebook—that contains the lesson as a project-based assessment.
- Links to helpful online resources.
- Job aids for writing outcomes and learning objectives (Appendix).

Lesson Planning Methodology: Understanding by Design

The lesson that we'll use as an example introduces students to *classification* and *simple clustering*, two important machine learning approaches. Before we go further, we'll define machine learning in a way that makes sense to us and recommend that you do the same for your students.

We built an instructional design framework for the sample lesson using an adaptation of *Understanding by Design* (UbD), a method created by Grant Wiggins and Jay McTighe (2005).

Many variations of the definition for machine learning exist, and you will want one that meets students where they are—as beginners, active practitioners, or experts. Here is one option from Vincent Granville. In his 2017 paper, “Difference Between Machine Learning, Data Science, AI, Deep learning, and Statistics,” he distinguishes between machine learning and AI like this:

Machine learning is a set of algorithms that train on a data set to make predictions or take actions in order to optimize some systems. For instance, supervised classification algorithms are used to classify potential clients into good or bad prospects, for loan purposes, based on historical data. The techniques involved, for a given task (e.g., supervised clustering), are varied: naive Bayes, SVM [support vector machine], neural nets, ensembles, association rules, decision trees, logistic regression, or a combination

All of this is a subset of data science. When these algorithms are automated, as in automated piloting or driver-less cars, it is called AI, and more specifically, deep learning.

If you wish, refer to Tharinda Paranagama's (2017) “An Introduction to Machine Learning,” for a great visualization of the components of machine learning.

You may very well have definitions that you prefer and that better suit your students. Share them with other Nexus members if you wish.

UbD advocates building lessons by first thinking about the end result (or outcome) in mind, and the definition that you choose will help you sculpt that result. Sometimes referred to as *backward design* or *assessment-based design*, UbD answers a fundamental question: “How do we (instructors) know that they (students) know?” Given the sometimes blackbox nature of machine learning and artificial intelligence, this is a great question and a great place to start.

Understanding by Design as a Lesson Integration Strategy

UbD is a useful framework for thinking more purposefully about planning your instruction and rebuilding any existing course modules to support a machine learning or data science lesson. UbD has a three-stage design process:

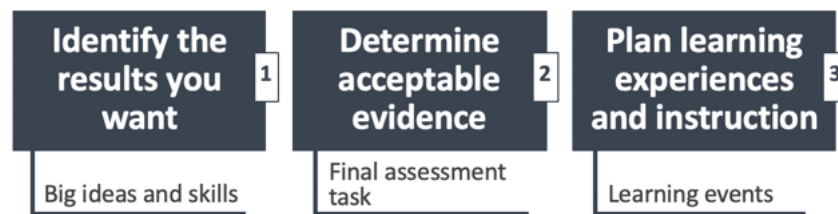


Figure 1. Backward Design
(adapted from Wiggins & McTighe, 2005)

Let's walk through the stages, which we've broken down into strategic questions, actions necessary to answer the questions, and supporting resources.

Stage 1: Identify the Results You Want

Strategy	Actions
Ask yourself the following questions: • What are the outcomes for my course or lesson module? • What do I want students to be able to know, understand, and do? • What are the essential questions for deep learning? How might I build this content into my lesson to achieve the outcomes?	To answer the questions, write outcomes and their supporting learning objectives to describe measurable, observable behavior of what the students will be able to do after completing this course or module. Use the outcomes to establish performance goals for the desired skills transfer. See the Appendix for two approaches to writing impactful objectives that will resonate with your students: • Audience, Behavior, Condition, and Degree of Proficiency (ABCD) • Specific, Measurable, Achievable, Relevant, and Timebound (SMART)

Stage 2: Determine Acceptable Evidence by Thinking Like an Assessor

Strategy	Actions
Ask yourself the following questions: • How will I assess or measure whether my students achieved the results I intended? • Per Wiggins & McTighe (2005): “What will we accept as evidence of student understanding and their ability to use their learning (transfer skills) in new situations? How will we evaluate student performance in fair and consistent ways?”	To answer to the questions, create assessment techniques for each supporting learning objective. Possible techniques include discussion, quizzes, and tests, of course, but student demos and projects may enable deeper learning. The following section reviews common assessment techniques (CATS) from Angelo & Cross (1993) to help you devise richer assessments.

Common Assessment Techniques and Activities

Most assessment techniques fall into these four categories:

Assessment Techniques for Evidence of Learning		
	KC	Knowledge checks/tests/quizzes
	D	Discussion/academic prompts
	SD	Student demonstration, teach backs
	PA	Project assignment

You are certainly quite familiar with all four. What we suggest is choosing which of the techniques best suits your particular learning objectives, with intention and attention, given that not all learning objectives are of equal weight, nor are assessments. To do this, you’ll want to unpack your priorities.

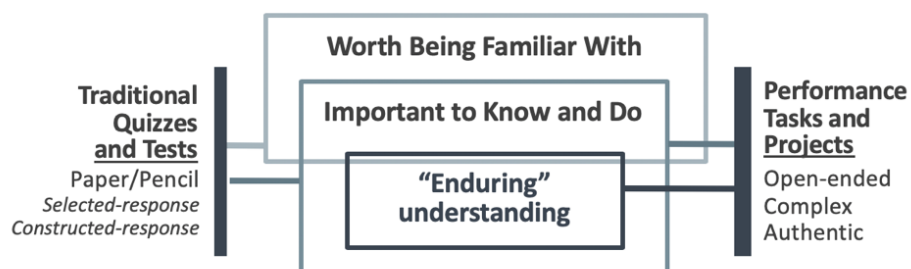


Figure 2. UbD Curricular Priorities
 (adapted from Wiggins & McTighe, 2005)

You can use the various assessment techniques from Angelo & Cross (1993) to apply as knowledge checks, discussion, student demos, and project assignments for a lesson that asks students to create a Jupyter Notebook. You'll see these assessments again in context later in this guide.

Stage 3: Plan Learning Experiences and Instruction

At this stage, you are providing content and tools for the students to be successful in performing the assessments we described in Stage 2.

Strategy	Actions
Start by asking these questions: - How will I support student learning and skills transfer? - What instruction, activities and resources will make students successful in completing the assessments and achieving desired results? - How will we sequence instruction?	To answer these questions, create course resources (presentations, articles, tools) and learning spaces (blogs, wikis, discussion forums) to support student success in performing the assessment.

Integrating a Machine Learning or Data Science Lesson into a Computer Science Course

What follows is a snapshot of one instructional design challenge for a 4-week module within a 15-week university-level course (see Figure 3 for a partial syllabus from Dr. Steve Scott's Computational Science Tools course, which is a featured course in the Generation AI Nexus).

		Points			
4	Python	Python basics, variables, expressions, strings, lists, formatted printing	Python Worksheet #1	9/23/2018	50
5	Python	tuples, dictionaries, functions, recursion	Python Worksheet #2	9/30/2018	50
6	Python	read/write data and CSV files, Objects and classes, Objects and inheritance	Python Worksheet #3	10/7/2018	50
7	Fall Break	No meetings	Nothing Due Free 50 points !	10/14/2018	50
8	Python	Matplotlib graphing, NetworkX graphs, Python applications, Jupyter Notebooks	Python Worksheet #4	10/21/2018	100
9	R	Intro to R, Intro to RStudio, variables	R Worksheet #1	10/28/2018	50

Figure 3. Partial Syllabus for Computational Science Tools Course

This module introduces machine learning and data science algorithms to produce meaningful visualizations. We chose it because it introduces *classification* and *simple clustering*, two important machine learning approaches. Classification and clustering enable practitioners to group data points together, the key difference being whether or not the groupings (i.e., classes) to be made are decided ahead of time. While these grouping techniques are a type of Artificial Intelligence designed to be performed by a computer, they can be used on any sort of data from any discipline.

Here are the instructional challenges:

- How might we extend Week 8 for this lesson without adding an additional week to the course?
- How might we ensure that the students will have adequate preparation in Python?
- How might we provide students time to practice, interpret, and reflect?

What follows is one way to adjust the design of this lesson.

Review Existing Module and Add Outcome

Let's assume that your current Python module curricular priorities map to three outcomes:

Students will be able to:	
1	Write executable code in Python
2	Apply object-oriented programming
3	Create data models with Python IDE

(Integrated Development Environment, a text and programming editor used by software developers. One popular IDE for Python developers is Spyder)

If you want to add a fourth outcome to the lesson, you'll need to make sure that you are not simply bolting on new tasking that doesn't lead to the deep learning outcome that you envision.

Students will be able to:	
1	Write executable code in Python
2	Apply object-oriented programming
3	Create data models with Python IDE
4	Create data models with Jupyter Notebook *NEW*

For this new outcome to serve your students well, we recommend that you ask yourself a series of questions that map to the actual outcomes you want for your class.

Ask Essential Questions for Each Outcome

As you re-evaluate the module alongside the outcomes your envision, ask yourself questions from a student's perspective—what does this professor expect me to know or learn? In UbD these are “Essential Questions” (Wiggins & McTighe, 2006, p.105) and are distinguished by intent and scope. Essential questions should be open-ended and go beyond just the topical; they should be “overarching” and invite deeper inquiry (“why” vs. “how”).

The questions below are more topical than overarching for this module because the intent and scope is skills development. However, per UbD, “Units and courses that focus on skill development *need to explicitly include* desired understandings. In other words, the learner should come to understand the skill's *underlying concepts*, *why* the skill is important, and what it helps to accomplish, what *strategies and techniques* maximize its effectiveness, and *when* to use them” (Wiggins, & McTighe, 2006, p.133).

Students will be able to:	
1	Write executable code in Python

- What are the basic data types in Python?
- How do I set up my development environment?
- How do I write executable expressions and lists?
- When do I work with tuples (a sequence in Python) and not lists?
- How do I work with dictionaries and functions?
- What are conditionals?
- What is recursion?
- How do I read in data, text?
- How do I write a data file?
- How do I get help?

Students will be able to:	
2	Apply object-oriented programming (OOP)

- How do I create my own data types?
- Why do we use methods?
- Is Python OOP?

- How do I handle exceptions and unforeseen results?

Students will be able to:	
3	Create data models with Python IDE

- How do I output my results?
- What are the steps to build a model using classes, methods, and constants?
- How can I make my code reusable with library modules and packages (.py extension)?
- How do I find and import libraries and packages?
- What's a good way to validate and report results?

Students will be able to:	
4	Create data models with Jupyter Notebook

- How can I use Python to explore data sets and build predictive models?
- What's a good way to validate and report results?

Use Essential Questions to Create Your Learning Objectives

The next step is to use the questions you just asked to create measurable learning objectives for each outcome. To this end, you will want to write learning objectives that support **Outcome 4** and that answer your questions.

Questions for Outcome 4: Create data models for Jupyter Notebook	Learning Objectives for Outcome 4
• How can I use Python to explore data sets and build predictive models? • What's a good way to validate and report results?	1. Analyze basic model parameters 2. Describe machine learning estimators and tasks 3. Create model for classification 4. Create model for regression 5. Create model for k-means clustering 6. Validate results (precision, recall, cross validation)

Your learning objectives should be written from your students' perspective – use action verbs per the cognitive domains of learning from “Six Facets of Understanding for Assessment” or Bloom’s taxonomy to layer your learning objectives. See Appendix for illustrations of both tools.

Map Assessments to Your Learning Objectives

How will you know that your students have achieved your learning objectives? How might you measure (assess) their learning?

Learning Objectives for Outcome 4	Assessments
1. Analyze basic model parameters (project assignment) 2. Describe machine learning estimators and tasks (knowledge check) 3. Create model for classification (project assignment) 4. Create model for regression (project assignment) 5. Create model for k-means clustering (project assignment) 6. Validate results (precision, recall, cross validation) (project assignment)	 Project Assignment: Write pseudo code for given “Diffusion of Innovation” model. Assign Rubric and Worksheet 4  Knowledge Check: Identify categories of learning problems: Supervised (classification, regression) and Unsupervised (clustering, density estimation); training v. testing data sets; compare data visualizations associated with each estimator    Project Assignment: • Create a Jupyter notebook • Explore IRIS data set (run classifier if necessary) • Run a simple clustering exercise • Run k-means clustering • Run a regression (train on 120 & test 30) • Interpret results and report recall & precision • Post results in Blog for comments, feedback, and discussion

In this example, you can make sure that your students will meet Learning Objectives 3-6 when they successfully complete the Project Assignment.

Add Content and Course Resources

You have now documented the objectives and outcomes. Now you need to provide or guide them to resources to be successful in completing the projects, knowledge checks, and discussions. Which instruction and media resources will help the students successfully complete their assessments and achieve your mutually desired outcome—passing your course with flying colors and being able to truly use the skills they’ve just practiced?

Assessments	Course Resources
 Project Assignment: Write pseudo code for given “Diffusion of Innovation” model. Use Rubric and Worksheet 4	• Readings: articles, lecture notes, videos • Websites • Discussion space • Data Sets • Tools: Jupyter Notebook • Job Aids/Worksheets • Grading rubric
 Knowledge Check: Identify categories of learning problems: Supervised (classification, regression) and Unsupervised (clustering, density estimation); training v. testing data sets; compare data visualizations associated with each estimator	
   Project Assignment: • Create a Jupyter notebook • Explore IRIS data set (run classifier if necessary) • Run a simple clustering exercise • Run k-means clustering • Run a regression (train on 120 & test 30) • Interpret results and report recall & precision • Post results in Blog for comments, feedback, and discussion	

You can embed your project assignment assessment into one interactive document—a computational notebook. The philosophy of lesson planning for machine learning and data science is to bring students closer to the data you want them to explore. Notebooks with embedded executable code facilitate the use of simple executable functions. Once you’ve written a script for a particular type of experiment or inquiry, it can be used repeatedly, even by students with little programming skill.

Delivering Instruction

Now that you have set the foundation for your module, you need a plan for delivering instruction. Your lesson pacing and sequencing of instruction will depend on a few things:

- Assessing prior learning of your students in terms of motivation and technical proficiency (KSA – knowledge, skills, abilities)
- Access to and availability of technology used in the lesson (e.g., Jupyter Notebook)
- Time limits and schedule of the course
- Classroom delivery modes: Classroom (in-person), online, blended

Stephen Brookfield offers many insights and practical suggestions to instructors. You can find many useful examples on lesson pacing and sequencing of instruction (*Lecturing Creatively*, Chapter 6, pp. 97-113) and on preparing for and moderating student discussions (Chapters 7 & 8, pp. 115-152).

ROPES Model

You may like the ROPES model as a useful instructional methodology for lesson pacing and sequencing instruction. ROPES (Review, Overview, Presentation, Exercise, Summary) enables an instructor to be more like a facilitator than the infamous “sage on the stage”. It is a way to introduce active learning in to your lessons. The Peak Performance Center website offers additional details (<http://thepeakperformancecenter.com/>).

Sample Lesson from Project Management Course

In this lesson, students are building team projects and need to modify their risk estimates using data science techniques. The data science module in this course is 2 weeks. Each week has a 1.5-hour live session, together supporting the following learning objectives:

1. Explore root causes of project failures using real-world data
2. Use data visualizations as a risk assessment tool
3. Modify risk estimates for student project using data-driven techniques

Below is a template for planning a live session using **ROPES**: **R**eview, **O**verview, **P**resentation, **E**xercise, **S**ummary.

Week 1 Session (time in minutes)	Learning Objective	Instructional Method	Activity + Media + Learning Resources
10	Review	Q&A & Presentation	Students present risk estimates for their projects
10	(1) Explore root causes of project failures using real-world data (2) Use data visualizations as a risk assessment tool (3) Modify risk estimates	(R) Question	Prompt for students to explain project vs. product risk calculations
10		(O) Discussion	Whole class discussion about defining risk estimates
10		(P) Whole class demonstration	Whole class demonstration of dataset for project failures; relate to project assignment
35		(E) Dataset exploration	Explore data set with Jupyter Notebook ; work with classifiers; create and test focused research questions; review job aids for Notebook
15		(S) Discussion	Clarify understanding of hypothesis testing; refine research questions

Checking for Success

One possible grading rubric for evaluating the performance of a machine learning project assignment is that of the Johns Hopkins Carey School of Business. We recommend it because it is thorough and nuanced enough for any lesson. Assessment criteria are

- Interpretation of data (qualitative)
- Analysis (quantitative)
- Critical evaluation of findings
- Ability to draw proper conclusions and make effective suggestions

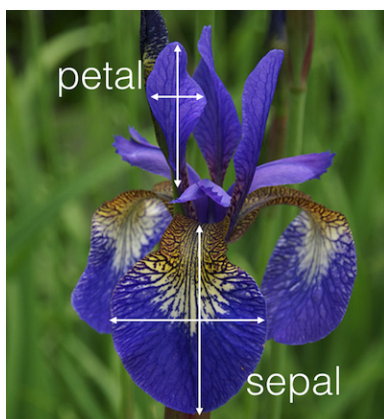
Rubric from Dr. Kyung-Soo Liew. (2017). Syllabus for BU.520.710, Big Data Machine Learning, Carey Business School, Johns Hopkins University. Reprinted with permission from the author.

Assessment Criteria	Not Good Enough (0 ≤ score < 6)	Good (6 ≤ score < 9)	Very Good (9 ≤ score ≤ 10)
Interpretation of Data (qualitative)	Little or no attempt to interpret data; or there are significant errors; or some data are over- or under-interpreted.	Interpret most data correctly; part of conclusions may be suspect; suggestions on future implementation are sound.	Data are completely and appropriately interpreted; there is no over- or under-interpretation; draw convincing conclusions.
Analysis (quantitative)	Methods are completely misapplied or applied but with significant errors or omissions. Choose inappropriate methods and make wrong predictions.	Most statistical methods are correctly applied but more could have been done with the data. Predictions are sensible but may deviate from the true results in a large range.	Statistical methods are fully and correctly applied; demonstrate superior data analysis skills; deeply mine the data and obtain useful insights for decision making.
Critical evaluation of findings	Blindly accept defective results; or recognize defective results but does not know how to fix them.	Recognize defective results and figure out the causes; understand the main sources of errors.	Show deep understanding for the sources of errors; recognize defective results and eliminates the causes
Ability to draw proper conclusions and make effective suggestions	Not draw conclusions; draw incorrect conclusions; suggestions are not acceptable.	Draw correct conclusion; suggestions may have potential impact on the future business.	Demonstrate substantial understanding of the problem; conduct deep data analytics using correct methods; draw correct conclusions with sufficient explanation and elaboration.

Sample Data Science Lesson in Simple Cluster Analysis

In this example, students will be using Jupyter Notebook for the first half of the project assignment:

- Create a Jupyter notebook
- Explore IRIS data set (run classifier if necessary)
- Run a simple clustering exercise – in this case, using the famous Iris Dataset, which addresses four features of this flower: sepal length, sepal width, petal length, and petal width.



The Iris Dataset uses the four attributes as columns to predict the class (i.e., species) of Iris. For this exercise, there are 3 classes of Irises: Setosa, Versicolor, and Virginica, and 50 sets of measurements for each class (150 data points in total). Following a previous lesson and assuming that the students have absorbed course materials (e.g., lecture, articles), this exercise aims to use the data set to learn (predict) which species of an Iris the student has found based on a set of attributes.

	sepal length	sepal width	petal length	petal width	target
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

To that end, here are two questions you might want to assign the students to answer:

1. Can we use these attributes to predict (distinguish) the class of Iris flowers? If yes, then which are best?
2. How can we improve our accuracy?

For this exercise, we are importing from a stable data source (the Iris Dataset), the data is clean, and it needs no transformation or imputation (for missing values). This makes the task easier than it would be if you had to curate the data for your students.

You can thus explain to your class that the task at hand is to list the features and attributes and visualize their relationships.

The first image and explanation they will see might be this (note that the instructor has provided lesson scaffolding in the form of comments and guidance in each cell):

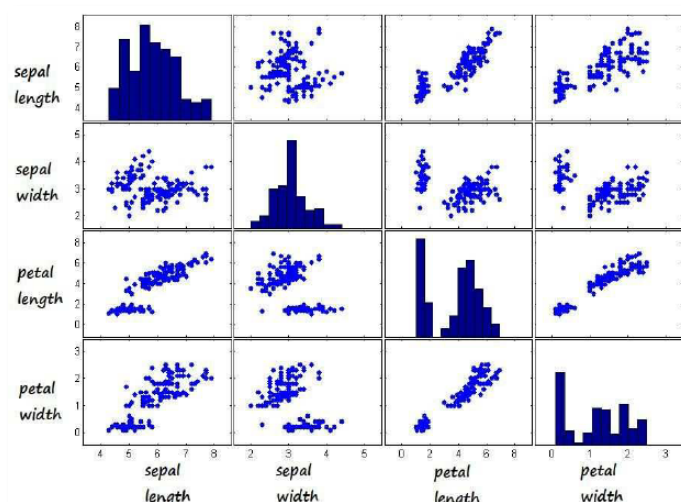


To accomplish our task of understanding the data, we will need to use some Python libraries. Some of the packages we will use today include [Pandas](#), [Scipy](#), [Numpy](#), and [MatPlotLib](#). If you want to learn more about how these different packages work with examples, click [here](#).

```
[2]: #  
# first, we need to import a bunch of Python libraries  
#  
# pandas is a very powerful data management package  
import pandas as pd  
from pandas.plotting import scatter_matrix  
# these are some very standard math/science Python libraries  
import scipy as sc  
import numpy as np  
import matplotlib  
import matplotlib.pyplot as plt  
# Scikit-learn is a widely used machine learning package  
import sklearn as sk  
from sklearn.datasets import load_iris  
[3]: #
```

Import Python libraries
& Iris data set

After a series of knowledge checks, you can move on to asking: What's the shape of the data? Here are two ways to visualize: scatterplot of the data and, for Setosa, descriptive statistics.



```
In [9]: #
# let's do some basic stats
# the data set has 50 observations per flower type
# so we can look at blocks of 50 rows at a time
#
for i in range(0,3):
    lo = 50*i
    hi = lo + 50
    name = dataset.target_names[i]
    print('data for flower type %s' % (name))
    print(pdata[lo:hi].describe())
```

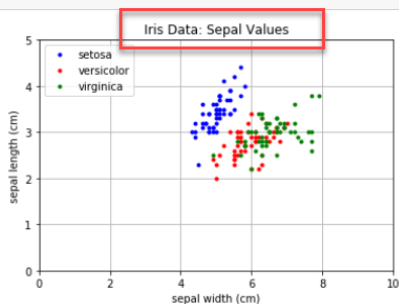
data for flower type setosa

	sepal length (cm)	sepal width (cm)	petal length (cm)	\
count	50.00000	50.00000	50.00000	
mean	5.00600	3.42800	1.46200	
std	0.35249	0.37906	0.17366	
min	4.30000	2.30000	1.00000	
25%	4.80000	3.20000	1.40000	
50%	5.00000	3.40000	1.50000	
75%	5.20000	3.67500	1.57500	
max	5.80000	4.40000	1.90000	

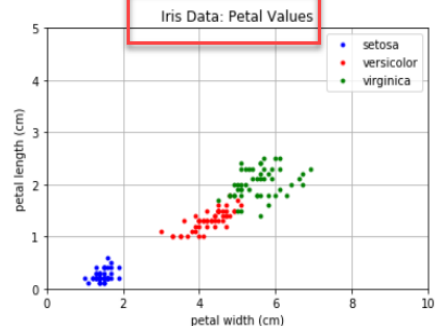
	petal width (cm)
count	50.00000
mean	0.24600
std	0.10538
min	0.10000
25%	0.20000
50%	0.20000
75%	0.30000
max	0.60000

If we explore paired attributes, which pairs will help us best see the clustering among the 3 classes?

```
#
# set up plot embellishments
plt.ylim(0,5)
plt.xlim(0,10)
plt.title('Iris Data: Sepal Values')
plt.ylabel('sepal length (cm)')
plt.xlabel('sepal width (cm)')
plt.grid()
plt.legend(('setosa', 'versicolor', 'virginica'))
plt.show()
```



```
plt.ylim(0,5)
plt.xlim(0,10)
plt.title('Iris Data: Petal Values')
plt.ylabel('petal length (cm)')
plt.xlabel('petal width (cm)')
plt.grid()
plt.legend(('setosa', 'versicolor', 'virginica'))
plt.show()
```



For this example, at this point in the lesson, a knowledge check for the students could be:

1. What is the mean petal size of each species of Iris?
2. What is the mean sepal size of each species of Iris?
3. Could you distinguish species using only sepal length and width? Why or why not?
4. Could you distinguish species using only petal length and width? Why or why not?

Online Resources for Jupyter Notebook, Python, and Machine Learning

To learn more about Jupyter Notebook and Python programming.

Quick introduction to Jupyter Notebook	https://www.youtube.com/watch?v=jZ952vChhuI
Project Jupyter	https://jupyter.org/
Python: YouTube video series	https://www.youtube.com/watch?v=k9TUPpGqYTo&list=PL-osiE80TeTskrapNbzxhwoFUiLCjGgY7&index=2
Python: Free Online Book	https://automatetheboringstuff.com/
Python: Online EdX Course	https://www.edx.org/course/introduction-to-python-absolute-beginner-3

To learn more about machine learning.

Intro to ML (first learn to code in Python then attempt this):	https://developers.google.com/machine-learning/crash-course/
scikit-learn: Machine Learning in Python	https://scikit-learn.org/stable/
Machine Learning Mindset: ML course in Python	https://github.com/machinelearningmindset/machine-learning-course

APPENDIX: Writing Impactful Objectives

A *learning objective* is a clear statement of (a) the skills or behaviors that you expect your learners to acquire and demonstrate during the course or module and (b) their level of performance.

Objectives are best understood as three levels in a hierarchy of outcomes:

- *Course outcomes* are those for the entire course, and tend to be expressed at a high level
- *Module outcomes* are for each lesson
- *Supporting learning objectives* ensure that the module outcomes are action-oriented and crafted from a student's perspective (i.e., what the learner will be able to do or perform as a result of their learning experience in your class)

As the instructor, you would determine achievement of these objectives with the assessment techniques we highlighted on pp. 7-8.

Six Facets of Understanding

Wiggins & McTighe (2005) explain that “Understanding is revealed when students autonomously transfer their learning through authentic performance. Six facets of understanding (i.e., the capacity to explain, interpret, apply, have perspective, empathize, and have self-knowledge) serve as indicators of understanding” (Wiggins, G., & McTighe, J. , 2006, p. 84). Their rubric, “When we truly understand, we can...” teases out what it means to be able to explain, interpret, apply, have perspective, empathize, and have self-knowledge.

Below is a summary of the cognitive domains of learning from the student's point of view and helpful action words that provide a stem to your learning objective statements.

When we truly understand, we can ...		
Explain	Generalize, connect, provide examples	communicate, debate, demonstrate, derive, describe, design, discover, exhibit, express, induce, instruct, justify, listen, model, predict, prove, read, synthesize, teach, write
Interpret	Tell stories, provide dimension	classify, communicate, draw conclusions, conclude, create analogies, critique, debate, discover, document, evaluate, investigate, illustrate, judge, make, meaning, make, sense, provide metaphors, interpret meaning, read, reason, speak, and write
Apply	Use what we know in real contexts	adapt, adjust, build, communicate, connect, create, debug, decide, design, estimate, invent, perform, produce, propose, read, solve, speak, test, use
Have Perspective	See points of view through critical eyes	analyze, argue, compare, communicate, contrast, critique, debate, infer, read, speak, write
Empathize	Walk in another's shoes, value what others do	be like, be open to, believe, communicate, consider, debate, imagine, read, relate, speak, write
Have self-knowledge	Access our metacognitive awareness, know that we don't know, reflect on meaning of the learning experience, self-assess	be aware of, communicate, debate, read, realize, recognize, reflect, self-assess, speak, write

Bloom's Taxonomy

Bloom (Fractus Learning, nd) offers another framework for mapping the cognitive domain.

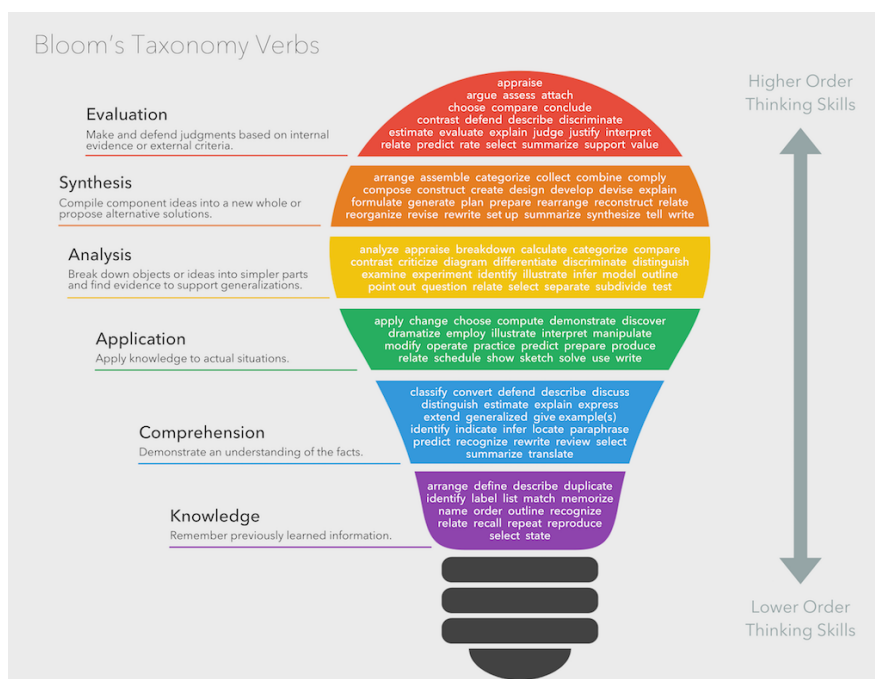


Figure 4. Bloom's Taxonomy

(Bloom's Taxonomy Verbs by Fractus Learning is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License)

We hope that this guidebook has been helpful to you, and we welcome your comments and suggestions. Write to us at generationainexus@mitre.org

References

Angelo, T., & Cross, K. P. (1993). *Classroom Assessment Techniques*. 2nd ed. Jossey-Bass.
Accessed: <https://cft.vanderbilt.edu/guides-sub-pages/cats/>

Brookfield, Stephen E. (2006). *The Skillful Teacher*. 2nd ed. Jossey-Bass.

Fractus Learning. (nd). Bloom's Taxonomy Verbs. Accessed:
<https://www.fractuslearning.com/blooms-taxonomy-verbs-free-chart/>

Granville, V. (2017). Difference Between Machine Learning, Data Science, AI, Deep Learning, and Statistics. Accessed: <https://www.datasciencecentral.com/profiles/blogs/difference-between-machine-learning-data-science-ai-deep-learning>

Liew, Kyung-Soo. (2017). Syllabus for BU.520.710, Big Data Machine Learning, Carey Business School, Johns Hopkins University. Accessed 6/10/2019 at
https://carey.jhu.edu/uploads/documents/520.710_AY17-18.docx

Paranagama, T. (2017). An Introduction to Machine Learning. Accessed:
<https://becominghuman.ai/an-introduction-to-machine-learning-7db04da817c4>

Peak Performance Center. (nd). ROPES. Accessed:
<http://thepeakperformancecenter.com/educational-learning/teaching-training/principles-of-instruction/ropes-model/>

Wiggins, G., & McTighe, J. (2006). *Understanding by Design*. 2nd ed. Alexandria, VA: ASCD.