



**Auburn University
Chemical Engineering**

Interactive Modules for Teaching Hands-on Data Science in Engineering

**Kerul Suthar, Thomas Mitchell, Anna Hartwig,
Jin Wang, Q. Peter He**

*Dept. of Chemical Engineering
Auburn University, Auburn, AL 36849*

- Gaps in data science education in Engineering
- Existing solutions and their limitations
- Proposed solution
- Examples

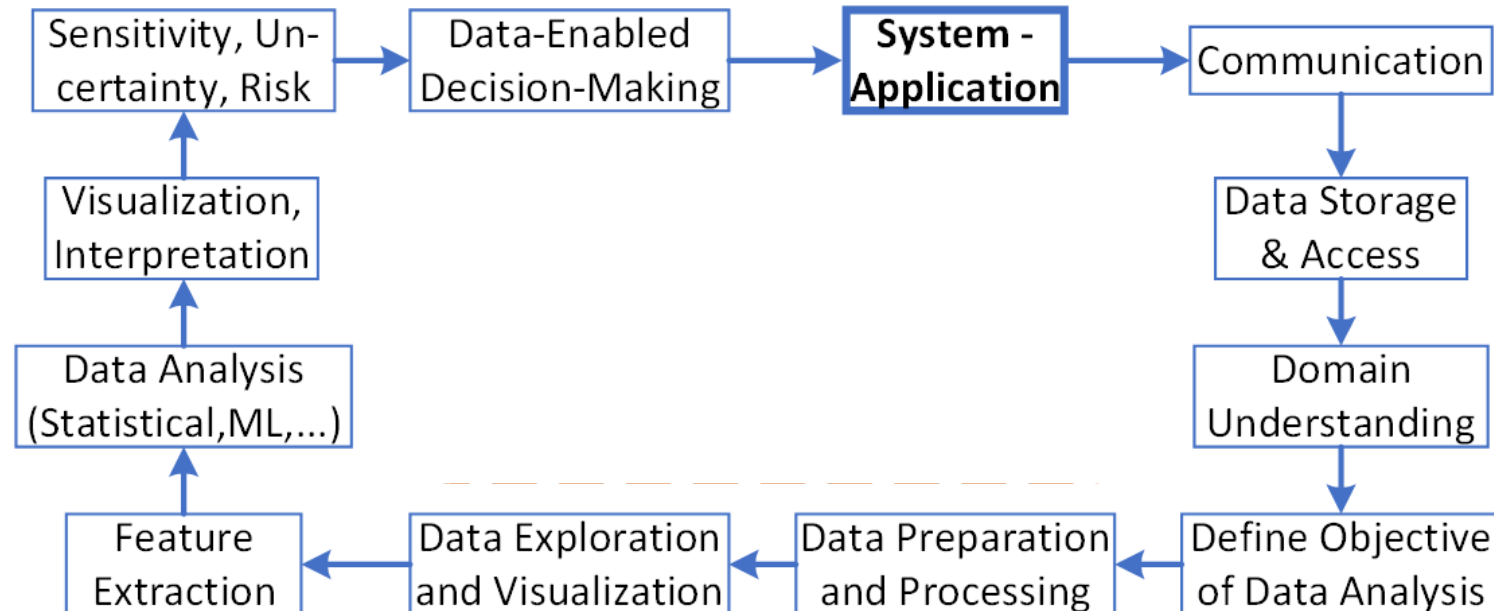
- Data science and engineering (DSE) education requires both appropriate classwork and hands-on experience with real data and real applications.
- Significant progress has been made in classwork
 - ▣ However, textbooks and traditional lecture courses offer limited help in developing students' capability in applying the theory and methods to solve real, complex problems
- Hands-on experience incorporating real data and real applications has been lacking
 - ▣ Students need practice on real data with the entire DSE cycle beginning with ill-posed questions and “messy” data¹.

¹ NASEM. *Data science for undergraduates: Opportunities and options*. 2018.

- Live or real project based DSE courses
- Real data based DSE course
- Other unconventional DSE courses
 - ▣ problem-based learning
 - ▣ learning by creating

- ❑ High burdens on instructor: time, organizational and pedagogical demands
- ❑ Solicitation of live projects is challenging
- ❑ Difficult to find data/application that motivate all students
- ❑ Some data may not have clear applications
- ❑ They do not generate learning materials that can be widely adopted at other institutions

- Data-enabled engineering project (DEEP) modules based on real data and applications
 - ▣ Cover the entire DSE lifecycle



- Data-enabled engineering project (DEEP) modules based on real data and applications
 - ▣ Serve as supplementary materials for DSE courses – replace or supplement textbook examples and homework problems to offer students hands-on experience
 - ▣ No curriculum change, minimize effort for adoption
 - ▣ Guided by experiential learning theory (ELT) – “learning through reflection on doing”¹

¹Kolb, David A. *Experiential learning: Experience as the source of learning and development*. FT press, 2014.

- ❑ Interactive Jupyter Notebooks using Python, R and SAS programming languages
 - ▣ Web-based interactive development and learning environment (IDLE) for easy adoption
- ❑ Hosted on Google Colab
 - ▣ Provide computation power
 - ▣ Provide software packages and libraries – Absolutely no local installation required (IDLE or any software/library) – All required is a web browser
- ❑ Embedded with critical and creative thinking questions
 - ▣ Learning through reflection on doing

□ Critical Thinking Questions for Module 1.2: Noise

▣ Remembering

- Have you encountered Fourier Transforms in any previous classes? How does it relate to filtering noise?
- Define noise in your own words. Will all data contain noise?

▣ Understanding

- What can cause noisy data? (A) Hardware failures (B) Errors made when processing the data (C) Difference between predicted and actual values (D) All of the above

▣ Applying

- What does the distance of smoothing tell us about the filters?
- How could noise reduction techniques be used in digital image processing?

▣ Analyzing

- What is the signal to noise ratio (SNR)? What does a high SNR mean? What does a low SNR mean?

- Critical Thinking Questions for Module 1.2: Noise
 - ▣ Evaluating
 - How does noise decrease the accuracy of machine learning? Can a signal get drowned out?
 - ▣ Creating
 - Design a plan to filter the noise of a linear dataset. What methods in this module could also be used for as a linear smoothing filter?

Features of DEEP modules (Python example)

The screenshot shows a Google Colab interface. At the top, the notebook title is 'Data Preparation and Processing - Missing Value.ipynb'. Below the title bar, there's a menu with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. A callout bubble points to the 'Easy to follow instructions' section of the notebook content. Another callout bubble points to the 'Connected to Google Cloud' status in the top right corner. The notebook content includes a section titled 'Data Preparation and Processing - Missing Values' with instructions for running the module on Google Colab. The instructions are as follows:

Instructions for running the module on Google Colab:

First click on the menu "File --> Save a copy in Drive". You will need to submit this file with answers to the embedded questions (in red) in this module.

Next click on "Connect" button on the right-hand side of the window toolbar.

Once connected, click on the menu "Runtime --> Run all", or Ctrl+F9 using your keyboard.

Click or double-click on places where you want to make changes or input your answers. These changes will be automatically saved to your copy of the Notebook.

Below the instructions, there's a section titled '1.1 Missing Values'.

Absolutely no local software/package installation required!

Embedded questions to ensure “learning through reflection on doing”

1.1 Missing Values

What do you know about missing values? (please type your answer below by double click on this cell)

1.1.1 Overview


Data cleaning is often the first step and one of the most important aspects of any data analysis project. The most common data cleaning is missing value detection and handling. There are many reasons that values may be missing from a dataset and these missing values can drastically affect data analysis such as regression or classification. In this module we will learn how to detect missing values in a dataset and later how to handle these missing values. Finally we will compare machine learning models using data with missing values and without missing values.

We will use the wood moisture data to show these principles. In this dataset, each row represents a sample while each column represent a feature or variable. The code underneath will serve as our import block so that all the imports needed are in an organized place. After that we will load the data that contains missing values. As shown below, this data contains many different types of missing values such as a string, NaN, and zero value. After we have confirmed that there is missing data we can move on to handling these missing values.

Which scenario is the most likely to have missing values occur?(please type your answer below by double click on this cell)

- A) A participant chooses not to respond to a question
- B) A participant drops out before a survey is completed
- C) A researcher makes a mistake in data entry
- D) A temporary failure of a temperature sensor
- E) All above

```
[ ] # Import Libraries
import scinv.io as spio # Load Data
```

```
frac = 0.0204  ;  
x_loess = smooth(t, x, frac, 'loess');  
  
plot(t, x, 'b. ');  
hold on;  
plot(t, x_loess, 'r-');
```

- How does the `frac` parameter affect the filtered data? What happens when the value is increased? What happens when it is decreased? Can it equal zero? Why or why not?

```
[b, a] = butter(3, high pass );  
x_butter = filter(b, a, x);  
  
plot(t, x, 'b. ');  
hold on;  
plot(t, x_butter, 'r- ');
```

- Why does the filtered data for the low pass Butterworth filter appear shifted to the right? What does the high pass Butterworth filter look like? What does this tell you about what the filter is doing?

 Open in Colab

In [2]: `!pip install saspy`

```
Collecting saspy
  Downloading https://files.pythonhosted.org/packages/86/72/06845b979f14e15994990bd06f4d662761bd0644c83f73f654b42a523a99/saspy-3.6.6.tar.gz (6.4MB)
    |████████████████████| 6.4MB 3.6MB/s
Building wheels for collected packages: saspy
  Building wheel for saspy (setup.py) ... done
  Created wheel for saspy: filename=saspy-3.6.6-cp37-none-any.whl size=6413279 sha256=9fed9d46bfa88ee03ec08318fdac88b4a500bd84b5a951611562ba327fe731fe
  Stored in directory: /root/.cache/pip/wheels/73/f5/fd/e50980b9187bdd695427d16f531ba2a57e44bc23c9041a2730
Successfully built saspy
Installing collected packages: saspy
Successfully installed saspy-3.6.6
```

In [3]: `import saspy`
`saspy`

Out[3]: `<module 'saspy' from '/usr/local/lib/python3.7/dist-packages/saspy/__init__.py'>`

In [4]: `sas = saspy.SASsession(java='/usr/bin/java', iomhost=['odaws01-usw2.oda.sas.com', 'odaws02-usw2.oda.sas.com', 'odaws03-usw2.oda.sas.com', 'odaws04-usw2.oda.sas.com'], iomport=8591, encoding='utf-8')`

```
Using SAS Config named: default
Please enter the IOM user id: kerulsuthar@gmail.com
Please enter the password for IOM user : .....
SAS Connection established. Subprocess id is 97
```

Exploring the bird dataset in SAS

It contains measurements on breeding pairs of land-bird species collected from 16 islands around Britain over the course of several decades. For each species, the data set contains an average time of extinction on those islands where it appeared, the average number of nesting pairs, the size of the species (large or small), and the migratory status of the species (migrant or resident). It is expected that species with larger numbers of nesting pairs will tend to be remain longer before becoming extinct.

Again, there is no local software/package installation required!

- Can be easily disseminated through web links

- DEEP-SM Module 1: Data preparation and processing

- ▣ <https://colab.research.google.com/drive/179xCiM3N9QAhhOSVFw2Cb90vmRXyBTl4?usp=sharing>

- DEEP-SM Module 2: Data Exploration and Visualization

- ▣ <https://colab.research.google.com/drive/14oVUgUW3A9ZNCCbhmyw6I2NrIFNdpsS?usp=sharing>

Planned DEEP module testing (2022)

Course	Instructor	Topics to be covered by DEEP modules
CHEN 6970 Big Data Analytics and Machine Learning in Process Industry	He (Auburn, Chem. Eng.)	Regression; Regression with regularization (Ridge, Lasso); Unsupervised learning (PCA, k-means); Supervised learning (PLS, CCA); Kernel methods; Support vector machines; Neural networks; Deep learning
ELEC 5110/6110: Wireless Networks	Mao (Auburn, ECE)	Communication systems; Communication protocols; Wireless access; Wireless LAN; Wi-Fi
COMP 5600/6600: Artificial Intelligence	Liu (Auburn, CS)	Clustering; Classification; Decision tree; Bayesian networks; Deep learning; Reinforcement learning
COMP 5630/6630/6636: Machine Learning	Liu (Auburn, CS)	Concepts, techniques, and applications in machine learning including abductive learning, case-based learning, deep learning, and reinforcement learning
STAT 4610: Applied Regression Analysis	Zeng (Auburn, Math & Statistics)	Simple linear regression; Multiple linear regression; Multicollinearity; Model selection and diagnosis
STAT 7650: Statistical Computing	Zeng (Auburn, Math & Statistics)	Linear regression; Regression with regularization (Lasso); Neural networks; Support vector machines; Decision tree; Random forest

- NSF-DUE (#1933873)
- DOE CESMII-EWD grant