Lecture 1 - Data Science, Statistics, and Machine Learning

Alon B. Muhame

September 30, 2024

Not every data science, machine learning and/or AI course is actually good enough for beginers. But, the implication or takeaway of most courses is (almost) always that a student will be a data scientist/machine learning expert and/or AI enginer or builder. While being able to work with data to extract insights and knowledge and apply it to solve a real-life problem (e.g., making a prediction of a phenomena – predicting sales in a particular period, churn for customers, analyzing text data etc) lies at the heart of data science/machine and/or AI exercise. [1]

[1] "We do not have knowledge of a thing until we have grasped its why, that is to say, its cause." – Aristotle

The goals in this lecture are:

- Enumerate tools and strategies to approach data science/machine learning and/or AI disciplines
- Emphasize a *multi-disciplinary* approach in studying these disciplines
- Set terminology/definitions for future discussions

Concretely, these set of notes aim to explain reasons why we study these disciplines and what are the best reading and writing advice:

1. develop our ability to understand and create data science, machine learning and AI arguments,
2. provide foundation to advanced data science/machine learning and AI courses,
3. generate better reading, learning and writing code best practices.

All of these topics and reasons are crucial in the development of your data science/machine learning and AI maturity. The importance of some of these concepts may not be apparant at the beginning. As time goes on, you will slowly understand why we include these courses in this program (Diploma in Computer Science Program) at WITI. In fact, you may not fully appreciate these subjects until you start start on your final projects, internship recess and even in the world of work.

An Overview

What is data science, statistics, machine learning and how are these concepts related . Roughly speaking, it is the study of data gener-

ating processes, extraction of insights, and using the knowledge to solve a real-life problem. Examples include (see notes in handout of this course)

This is a challenging course partly because of its intensity. We have to cover many topics that appear totally unrelated at first. This is also the first time many students (like you) have to study fields/disciplines that make of the subject matter (esp. science subjects matter) in application. You will be asked to organize your thinking around solving a problem leveraging data clearly, precisely, and rigorously, which is a new experience for most of you.

Learning how to think and apply especially concepts so far studied in your education journey (from esp. A'level) is far more important than knowing how to do all the computations, or remember all the formulas etc. Cosequently, the principal objective of this course is to help you develop analytic skills you need to use tools such as Excel/Python etc and concepts you need to learn data science/machine learning and AI. To achive this goal, we will show you the motivation behind these ideas, explain the results, and dissect why some of the methods work or may not work in real practice and what are the pros and cons of these methods in the real world settings.

Suggestions to Students

All computer science (software coding disciplines) are difficult. It takes hard work and patience to learn these concepts. Rote memorixation (crammer work) does *not* work.

Comment 1

Here are some suggestions that you may find helpful. Thses could some of the many things to succeed in Computer science:

1. *Do not skip classes*
2. *Read the text, including the examples, before the lecture; review what you have learned after each lecture.*
3. *Do the exercises (assignments- lab (coding) projects)*
   - *a: First, study the examples in the notes*
   - *b: Make an effort to understand how and why a code block or section works, and remember how certain types of problems should be solved.*
   - *c: When you do a problem, ask yourself if you have seen somthing similar before; if you have, follow the steps in its solution (could be in this course or other units in this program)*
   - *d: After solving a problem, look for alternate solutions, anlayze and compare their difference.*
4. *Get help from the instructors, your friends, and whatever facility you institute or school provides.*
5. *Develop good study habit.*
   - *a: Keep working every day; study the notes(handouts/slides/code files/code notebooks, etc), your own lecture notes, and most important of all, do the lab coding practices at the end of each section or class.*
   - *b: Form a study group of two or three students, and meet on a regular basis to study togther.*
   - *c: Check the solutions (lab coding practices - quizzes) for any nonsense or discrepancies*
   - *d: Learn how to solve the problems systematically - computer science entails systematic thinking and approach.*
6. *Perseverance. Do not give up easily.*
7. *Be willing to help your classmates. Trying to explain something to others is the best way to learn anything new.*

Attitude is the real difference between success and failure

Nothing comes easy. To succeed, you have to work hard. But you also need to learn how to learn coding or generally computer science (e.g., data science) the right way.

- Do not rely on memorizing formulas or procedures by rote. Instead, try to understand the concepts and ideas behind them. It is important to learn when and how to use them.
- Of course, it does not mean that you need not to memorize anything at all. On the contrary, many basic results and definitions and conventions (e.g., python libraries etc) need to be memorized. You may find it helpful to use a highlighter to mark definitions and keywords that you have trouble recalling, and I urge you to review them frequently.
- Do not compartmentalize the materials in this program; all the course units are connected in one way or another. Consequently, as you move along from course unit to course unit, semester to semester, try to observe the connections between the concepts you have learned. Without saying, it is understood that you need to remember what youbhad learned earlier or in a different course unit.
- Take you homework assignments seriously. Keep in mind: study for a test, you may want to review your homework, so you need to be able to your own work. Write everything clearly and neatly. The process of writing out everything correctly helps think about what you write. Very often, incohorent and incomprehensible writing is an indication of lack of understanding of the subject matter.
- When doing your homework assignments, start with a draft, then look over it carefully, check the spelling and grammar, and revise the solution. Make sure you write in complete sentences and use correct notations. If neccessary, you may have to polish it further. Before turning in the final version, be sure to check again for any mistakes that you may have overlooked.

How to Read and Write Computer Programming Code

Reading and writing code is difficult for beginners. It takes patience and practice to learn how to read, write and understand computer code or programs. You may need to read a piece of code or sentence or a paragraph several times before you understand it completely. There are writing styles and notational conventions that you acquire

only by reading and paying attention to how computer code or programs are written. As we proceed with the course, we will discuss the details. As a starter, let us offer several suggestions.

- Make sure you know the definition (symbols) of computer programs, the meaning and proper usage of the computer symbols and notations. Although this may sound obvious, many beginners have difficulty understanding a computer program argument because they fail to recall the exact meaning of certain computer program concepts.
- Often, the reason behind a program or piece of code lies in the structure or object class or code before it. Sometimes it could be found in the preceeding paragraph, and it it not unusual that you may need to check several sentences or paragraphs before it. You need to take an active role in reading computer programs and you need to remember what you have read.
- Computer programs (data scientists inclusive) prefer shorts and elegant notations. To do this, they suppress the details of what they consider as "obvious" reasons. But what is obvious to one reader may not be that obvious to another. At any rate, for practical reasons, it it impossible to include every minute step in a computer program or piece of code. Consequently, keep your pen and notebook next to you, and be ready to check code and fill in the missing details for your understanding.

Writing Computer programs or code even harder! It takes much longer to learn how to write programming code. Of course, the most important thing about a computer science or data science program is its correctness (after all it will not produce any output – it will produce an error). When we say "good" computer code or program, we are talking about precision, clarity, and sound logic[2].

[2] every bug is a lesson and an error is an opportunity to grow