Supplemental Material: Annu. Rev. Stat. Appl. 2017. 4:15–30
http://doi.org/10.1146/annurev-statistics-060116-053930
Curriculum Guidelines for Undergraduate Programs in Data Science
De Veaux et al.

8. APPENDIX—DETAILED COURSES FOR A PROPOSED DATA SCIENCE MAJOR

1. **Introduction to Data Science I**

   (a) Vision

   - A complete alpha-to-omega introduction to data science. Students will engage in the full data workflow, including collaborative data science projects. This class is meant to be a high-level introduction to the spectrum of data science topics, probably best taught in an iterative cycle from initial investigation and data acquisition to the communication of final results

   (b) Learning goals

   - Exploring and wrangling data
   - Writing basic functions and coding
   - Summarizing, visualizing, and analyzing data
   - Modeling and simulating deterministic and stochastic phenomena
   - Presenting the results of a complete project in written, oral, and graphical forms

   (c) Topics

   - What are data?
     - What forms do they take? What sorts of questions can they answer?
       * Description, prediction, inference
       * Implementation: Data should be nontrivially complex, but relatively clean. At this stage the data should be given to students and should be sufficiently rich in terms of number of variables, size, multiple tables, etc.
   - Introduction to high-level programming language and integrated development environment (IDE) (R recommended)
   - Describing data: exploratory data analysis (EDA) and data visualization
     - Summaries, aggregation, smoothing, distributions
   - Modeling and stochastics (understand notions of uncertainty, simulations, random number generator, etc.)
     - Notion of mathematical model or function (e.g., linear, exponential) and programming concepts
     - Vectors, tables/data frames, variables
     - Sequential programming/scripting
     - Defining very simple functions, basic environment, and scoping rules
     - Conditional expressions and basic iteration
   - Simulation with/without data: probabilistic and/or resampling based
   - Algorithms
     - Breaking a complex problem down into small steps
     - Concept of an algorithm as a recipe
   - Project

- Communication: evaluate students on written, oral, and graphical forms
- Ethics: e.g., copying code is plagiarism
- Teamwork: working on a group project with version control (e.g., GitHub). Concepts of project and code management.
- Motivate with application areas

2. **Introduction to Data Science II**

(a) Vision

- Exposure to different data types and sources, the process of data curation for the purpose of transforming them to a format suitable for analysis. Introduction to the elementary notions in estimation, prediction and inference. We envision this class to be taught through case studies involving less-manicured data to enhance their computational and analytical abilities.

(b) Learning goals

- Interacting with a variety of data sources including relational databases
- Accessing data via different interfaces
- Building structure from a variety of data forms to enable analysis
- Formulating problems and bringing elementary concepts in estimation, prediction, and inference to bear
- Understanding how the data collection process influences the scope of inference

(c) Topics

- Kinds of data: e.g., static, spatial, temporal, text, media, etc.
- Data sources: e.g., relational databases, web/application programming interface (API), streaming
- Data collection: e.g., sampling, design (observational versus experimental) and its impact on visualization, modeling and generalizability of results
- Data cleaning/extraction: e.g., wrangling, regular expressions, SQL statements
- Data analysis/modeling:
  - Question/problem formation along with EDA
  - Introduction to estimation and inference (testing and confidence intervals) including simulation and resampling
  - Scope of inference
  - Assessment and selection e.g., training and testing sets
- Project: A more comprehensive data-driven project including problem formulation, informed data wrangling and elementary analysis and conclusions including limitations to generalizability
  - Ability to collect data is enhanced
  - Better tools for analysis/scope of inference
  - Communicate, team, ethics, presentation

3. **Mathematics for Data Science I**

(a) Vision

- Data science students need to use mathematical and statistical models, and these require mathematical foundations. This course introduces concepts from linear modeling and optimization, providing mathematical foundations as they are needed and motivated by applications. The focus is not on proof nor on excessive hand computations; instead, it is on employing and relating the mathematics to the real-world ideas. Concepts are made concrete through numerical computation.

(b) Learning goals

- Model real-world phenomena with functions (e.g., linear, polynomial, exponential)
- Optimize functions and interpret results
- Develop geometric intuition for linear modeling and optimization
- Employ tools for understanding local behavior of functions and models
- Learn tools from matrix algebra to analyze higher dimensional models
- Investigate sensitivity of models to small changes in the inputs
- Navigate the choice of model; understanding limitations of models
- Generate models for varied applications, using a diversity of tools

(c) Topics

- Introduction to modeling: simple models via functions (e.g., from physics)
- Linear Algebra and Geometry: vectors, dot product, distance, projection, matrix algebra
- Multivariate thinking: (hyper)plane equations, systems of equations
- Sensitivity of models to changes in the input: slopes of lines, cross sections of planes, secant and tangent lines for curves
- Solving linear systems of equations:
  - Geometric approach to the determinant as a measurement of volume
  - Row operations, echelon form, and geometric interpretation
  - Degenerate matrices, nonexistence of solutions, criterion for existence of solutions
- Derivatives:
  - Linear approximation
  - Optimization in the context of data science and regression
  - Gradient method of steepest descent
- Geometric approach to correlation via the dot product
- Matrix factorizations (if time allows)
  - LU and QR factorizations
  - Geometric interpretation of linear transformations
  - Eigenvalues and eigenvectors, diagonalization
  - Singular value decomposition
  - Principal component analysis

4. **Mathematics for Data Science II**
   (a) Vision
      - Introduce probability theory and data-generating processes that lead to probability distributions. Build probability distributions from empirical data, and density functions from histograms. Topics of integration are introduced as motivated by probabilistic ideas and the transition from discrete data to continuous functions. The focus is not on proof nor on excessive hand computations; instead, ideas and concepts are made concrete through visualization and numerical computation.

   (b) Learning goals
      - Understand and appreciate the role of randomness in data-generating processes
      - Apply properties and rules of probability to specific problems or questions
      - Use simulation at a basic level to generate data and to explore probability
      - Interpret area under a curve in a probabilistic context
      - Identify appropriate use of probability distributions, including ideas of expectation and variance
      - Extend univariate concepts to their multivariate analogues

   (c) Topics
      - Randomness, random variables, empirical distributions
      - Notions of probability: sample spaces and events, sets and counting, Venn diagrams
      - Simulation as a constructive tool to explore probability, law of large numbers
      - Conditional probability and independence: Bayes' formula
      - Binomial, geometric, and normal distributions. Others taught "just in time" as needed in advanced courses
      - Integration
        – Motivate integrals via probability, expectation, and variance (discrete to continuous)
        – The fundamental theorem of calculus: relationship between probability distribution function and cumulative distribution function
        – Understanding basic numerical integration and the idea of error bounds
      - Sampling distributions and the central limit theorem through simulation
      - Joint probability distributions, multiple integrals, marginal distributions
      - Series and connection to discrete probability distributions

5. **Algorithms and Software Concepts**
   (a) Vision
      - Deepen students' understanding of computational problem solving begun in Data Science I and II. Continued use of a data-centered approach and employing scaffolding of problems and techniques, using procedural programming and exploring common problem-solving strategies and assessing efficiency. Utilize statistical algorithms and data methods as a vehicle for learning the computer science concepts.

(b) Learning goals

- Writing complete programs to solve problems
- Learn a second programming language (e.g., Python, C++, Java)
- Gain comfort with a new IDE (e.g., Eclipse)
- Deepen programming language understanding, including concepts of variable scoping, data types, parameter passing, etc.
- Understand implementations of basic statistical algorithms and their efficiency, from both abstract and lower level perspectives

(c) Topics

- Applying fundamental data structures useful in a data science context
  - Lists (function application, aggregating statistical properties)
  - Stacks (encoding sequential data transformations)
  - Queues (simulating arrival processes)
  - Dictionaries (implementing graph representations, bag of words)
  - Matrices (implementing factorization algorithms, implementing group-by/summarize)
  - Trees (implementing recursively partitioned decision tree)
  - Efficiency of operations applied to the above data structures
  - Scalability of various data structures, including distribution beyond a single node
- Complexity of algorithms
  - Big-O notation, practical versus theoretical efficiency
  - Brute force, divide-and-conquer
- Software development process
  - Correctness, evaluation, debugging
  - Code readability/documentation
  - Project and code management—version control
- Comparing programming languages
  - Functional programming centered view from Data Science I and II versus imperative programming: the value of provable correctness, the ability to map an operation across an entire list, connection to system design, ethical implications
  - Basics of object-oriented programming: software design using abstract data types
- High-performance computing
  - Algorithmic paradigms (e.g., map-reduce, online/streaming)
  - Systems (e.g., Hadoop, Spark, NoSQL)

6. **Data Curation, Management, Organization**

(a) Vision

- Deepen students' knowledge of data curation and management, building from the foundation of Data Science II by acquiring data across a spectrum of single and multiple sources and varying degrees of structure, incorporating real-world data throughout this spectrum. Gain understanding in how to transform and wrangle acquired data into forms amenable for analysis. Extend data management concepts and skills to accommodate big data.

(b) Learning goals

- Acquire data from a spectrum of external systems, ranging from well-structured systems with defined schema to unstructured, requiring access through APIs and requiring scraping/munging, to large-scaled distributed storage systems
  - Acquire data from multiple sources. Be able to organize and unify them
  - Be able to learn to learn—generalize to new APIs and new systems and be able to acquire data from them
  - Understand the statistical issues inherent in the sources of data—bias, randomness or lack thereof—and the impact of those issues on analysis and generalization
- Understand the structure of data needed to enable exploration, visualization, and analysis, and be able to take data not in that form and transform it appropriately
  - Advanced data cleaning and transformation
  - Blend tools from a variety of sources to combine and analyze data

(c) Topics

- Query languages and operations to specify and transform data (e.g., projection, selection, join, aggregate/group, summarize)
- Structured/schema based systems as users and acquirers of data
  - Relational (SQL) databases, APIs and programmatic access, indexing
  - XML and XPath, APIs for accessing and querying structured data contained therein
- Semistructured systems as users and acquirers of data
  - Access through APIs yielding JSON (JavaScript Object Notation) to be parsed and structured
- Unstructured systems in the acquisition and structuring of data
  - Web scraping
  - Text/string parsing/processing to give structure
- Security and ethical considerations in relation to authenticating and authorizing access to data on remote systems
- Software development tools (e.g., GitHub, version control)
- Large-scale data systems
  - Paradigms for distributed data storage
  - Practical access to example systems (e.g., MongoDB, HBase, NoSQL systems)

        * Amazon Web Services (AWS) provides public data sets in Landsat, genomics, multimedia

7. **Introduction to Statistical Models**
   (a) Vision
      - Introduces students to a framework for inference using regression models. The foundation is linear models, which is then compared to nonlinear approaches. The course builds on important concepts introduced in the first-year data science courses that form the foundation of any statistical analysis. All the ideas are firmly grounded and inspired from real-world data. A sound understanding of the ideas in this course will enable the student to understand the common structure underlying important algorithms and methods in Statistical and Machine Learning.

   (b) Learning goals
      - Understand the mathematical framework for linear models and their applications
      - Apply regression models to real data sets for the purpose of scientific inference, understanding and communication
      - Introduce both theoretical and simulation approaches to uncertainty in models

   (c) Topics
      - Review of hypothesis testing, confidence intervals, etc.
      - Estimation (e.g., likelihood principle, Bayes)
      - Linear models
        – Regression theory (i.e., least-squares): introduction to estimation principles
        – Multiple regression
          * Transformations, model selection
          * Interactions, indicator variables, ANOVA
        – Generalized linear models e.g., logistic, etc.
      - Alternatives to classical regression e.g., trees, smoothing/splines
      - Introduction to model selection
        – Regularization, bias/variance tradeoff (e.g., parsimony, AIC, BIC)
        – Cross validation
        – Ridge regressions and penalized regression (e.g., lasso)

8. **Statistical and Machine Learning**
   (a) Vision
      - This is a course that blends the algorithmic perspective of machine learning in computer science and the predictive perspective of statistical thinking. It will focus on the common machine learning methods and their application to problems in various disciplines. The student will not only gain an understanding of the theoretical foundation of statistical learning, but the practical skills necessary for their successful application to new problems in science and industry.

(b) Learning goals

- Gain experience with specific, widely used machine learning algorithms
- Understand the basic theoretical underpinnings of the methods and their limitations in inference
- Be able to apply the methods to real data, evaluate their performance, and effectively communicate the results

(c) Topics

- Algorithms—what is the process doing (input/output), what can this solve.
  - Practical issues of model implementation (e.g., scalability, curse of dimensionality)
- Performance metrics and prediction
  - Loss functions
  - Model selection and assessment, e.g., cross-validation, performance measures, regularization methods
- Data transformations
  - Dimension reduction, principal component analysis, feature extraction
  - Smoothing and aggregating: e.g., kernel methods, spatial and/or temporal smoothing, model averaging
- Supervised learning
  - Regression: linear models, regression trees
  - Classification: classification trees, logistic reg, separating hyperplanes, k-nearest neighbor (KNN) methods
- Unsupervised learning
  - Clustering: k-means, hierarchical
- Ensemble methods (e.g., boosting and bagging)

(d) Project (Common Task Framework)

9. **Capstone Course**

(a) Vision

- The capstone course provides students with a comprehensive learning experience that integrates knowledge from previous courses. The student will make connections among ideas and experiences gained from all three core disciplines and apply them to another domain. In the course students will synthesize what they have learned and apply that knowledge to new, complex situations. Student should engage in the entire process of solving a real-world data science project, from collecting and processing actual data, to applying suitable and appropriate analytic methods to the problem, and communicating the results in a clear and comprehensible way. The capstone should emphasize a good understanding of the foundational knowledge of the core disciplines and the domain area, and prepares the student for future professional endeavors.

(b) Learning goals

- Ability to handle a problem in data science from the initial problem formulation through a proposed solution and then communicating the results of that effort. The student will demonstrate proficiency in the entire process of collecting, preparing and analyzing real-world data to provide a solution to the initial problem.
- Learn how to work in teams by working with at least one other student on their project
- Present results and code which are reproducible and ethical
- Present a report (both oral and written) on the project emphasizing
  i. Motivation and problem definition, existing approaches to the problem
  ii. Proposed solution
  iii. Results, conclusions, limits to the present analysis, and directions for future work

(c) Suggested areas and topics for capstone projects:

- **Application project.** A project mainly focused on an application of interest to the student. The student should explore how best to apply learning algorithms to solve the problem of interest, and be able to evaluate the performance and the limitations of the model.
- **Algorithmic/computational project.** Motivated either by existing algorithms and the literature pertaining to them, or from an application area, a student may propose a new learning algorithm, a novel variant of an existing algorithm, or a novel application of an existing algorithm.
- **Theoretical project.** For the more ambitious student, analyzing and possibly proving some properties of a new or an existing learning algorithm.