Section 1: Week 1: Evaluate Tools for Statistical Applications

Nate Bachmeier

TIM-7101: Statistics for Technology Leaders

August 9, 2020

North Central University

# Evaluate Tools for Statistical Applications

When a project requires significant investments into statistical modeling, it can quickly become prohibitively expensive and tedious to perform these calculations by hand. Instead, the analyst must defer to software solutions to transform raw data values into business intelligence. Luckily there is a vast ecosystem of tooling that can specialize in different scenarios, such as interactive ‘slicing and dicing’ versus offline batch processing. An organization like NCU-C also must consider how the various products align with existing data platforms. For instance, a business that relies on traditional relational data stores might have more flexibility than another institution that requires graph technologies. Just as remodeling a kitchen necessitates hammers, screwdrivers, and tape measures—statistical applications can involve multiple tools. Consider the situation where data begins life in unstructured data lakes, and through an extract transform and load (ETL) process becomes a geographical map. This situation might call for Python scripts to parse records into a tabular format. Next, confirming the dataset is complete can call upon programs like Microsoft Excel, IBM SPSS, or Tableau to visually create pivot tables and charts. Finally, importing the geospatial data into software like QGIS provides a canvas for further domain-specific explorations.

# Programming Interfaces

## Static versus Dynamic

While any programming language can perform statistical calculations, there are inherent advantages that make one a more natural choice over another in specific-contexts. For instance, when a project begins, the requirements are more nebulous, and this shifts the focus to developer efficiencies over runtime performance. During this initial period, scripting languages like R and Python allow the analyst to experiment and find the best modeling technique. Later, porting these scripts to C++ and Fortran increases the cost for additional modifications but provides hardware-native execution speeds. These low-level languages also enable controls over memory management, remote procedure calls, and local thread scheduling. Engineering teams can use these primitives to scale-out systems to enormous datasets and related High-Performance Computing (HPC) tasks.

## Matrix versus Scalar Models

Performing data transformations across large quantities of data will require significant computational resources. These calculations must happen efficiently through matrix and vector processing, not general-purpose loop-constructs. For instance, iterating through a list of objects can encounter much overhead in accounting for index offsets. Meanwhile, data frames libraries are specifically optimized to apply manipulations on high-dimensional data. These optimizations enable packages to make several updates at the same time or scale-out distinct partitions across distinct processing units, such as Central Processing Units (CPU), General Purpose Graphics Processing Units (GPGU/GPU), Application-specific Integrated Circuits (ASIC), and Field-Programmable Gate Arrays (FPGA).

# Interactive Software Packages