Financial Data Science with SAS®

Babatunde O. Odusami

The correct bibliographic citation for this manual is as follows: Odusami, Babatunde O. 2024. *Financial Data Science with SAS®*. Cary, NC: SAS Institute Inc.

Financial Data Science with SAS®

ISBN 978-1-68580-013-0 (Paperback)
ISBN 978-1-68580-014-7 (Web PDF)
ISBN 978-1-68580-015-4 (EPUB)
ISBN 978-1-68580-016-1 (Kindle)

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

June 2024

Contents

About This Book

What Does This Book Cover?

The use of data science techniques such as machine learning, data visualization, and optimization is widespread in the financial services industry. However, it is very hard to find books and reference materials that explain the theory behind these techniques and how to implement them using real-world examples. There are also not that many instructional and reference materials for those interested in using SAS for financial data science, although SAS is arguably one of the best applications for tackling pretty much any type of analytics problems you might encounter in the finance domain, as you will see shortly in the book.

The book will provide readers with a comprehensive coverage of the theoretical and practical implementation of the various types of analytical techniques and quantitative tools that are used in the financial services industry using SAS as the main analytics platform. It will show you how to implement data visualization, simulation, statistical predictive models, machine learning models, and financial optimizations using real-world examples in the SAS analytics environment.

You will learn how to use visualizations to measure financial performance and examine the salient characteristics of financial and economic data. You will also learn how to implement various types of simulations and how to use simulations to build models of financial data, such as stock prices and capital project outcomes. You will be introduced to advanced applications of simulations, such as using simulations for risk management and pricing of derivatives contracts.

In subsequent chapters, you will be introduced to various statistical and machine-learning models and how they are used in the financial services industry. These include credit risk models, algorithmic trading models, economic analysis models, and strategic portfolio management models. You will also enhance your analytics skills with practice exercises that apply these models to real-world data.

In the last section of the book, you will learn about the various types of optimization techniques and how they are used to solve decision problems in finance. You will review and practice the use of optimization for capital budgeting, profit maximization, risk management, performance attribution, portfolio construction, and portfolio optimization.

The book is a reference material for anyone interested in learning more about how data science is used in the financial services industry and how to develop data science competencies in finance using the SAS Analytics suite.

Is This Book for You?

If you are interested in building competencies that will prepare you for career roles such as investment analyst, quantitative analyst, or financial data scientist, or just interested in learning more about how to use data science methods to improve your financial skills, then this book is for you. This book aims to make the data science journey in finance a much easier experience for everyone, both seasoned and aspiring data scientists. It is also a great handbook for data science generalists who would like to transition into financial data science. The book can be used for a semester-long course in financial data science for graduate or advanced undergraduate students in finance, mathematics, business analytics, and computer science programs.

SAS Programming Experience

Minimal SAS experience is required. You will be introduced to the fundamentals of SAS programming and the SAS Windowing environment in the early chapters before we delve into some of the more advanced topics. Some familiarity with programming logic and general finance and economic concepts (such as stocks, bonds, stock index, returns, risk, portfolios, income, capital projects, GDP, and unemployment) will be helpful. Most of the advanced concepts are also summarized in plain language, so you can focus your attention on the practical applications of the techniques that will be discussed. The book will also point you to where you can find additional resources from the vast repository of SAS support documentation and on the internet.

What Should You Know About the Examples?

This book includes many tutorials for you to follow so that you can gain hands-on experience with using SAS for solving a wide range of analytical problems that you might come across in the financial services industry. In many cases, the links between the statistical and economic theories and the SAS code in the book are highlighted so that you can see the connections between them. You will learn about the code, but more importantly why specific code or methods are relevant from the theoretical point of view.

Software Used to Develop the Book's Content

To get the full experience of the tutorials in the chapters and to complete the practice exercises, you will need access to the SAS 9.4 environment. You can access the SAS 9.4 environment using client applications such as SAS Studio, Enterprise Guide, and Enterprise Miner. The SAS 9.4 environment includes the Base SAS software. You will also need a license for SAS Analytics 15.3, which includes:

- SAS/ETS
- SAS/STAT

- SAS/IML
- SAS/OR
- SAS Enterprise Miner

If you are a student or employee of an academic institution, you can sign up for access to SAS OnDemand for Academics at https://welcome.oda.sas.com, which will give you access to all of the features above in a cloud-enabled environment. Independent learners can also sign up for SAS OnDemand for Academics for access to SAS Studio.

Example Code and Data

You can access all the SAS code and data for this book by visiting the book's GitHub page at https://github.com/finsasdata/bookdata.

In the GitHub repository, you will find two SAS programs that will enable you to easily download the data that you need for the tutorials and practice exercises. The first program (GitDownload.sas) will download the entire repository into a SAS library called FINDATA. The second program (Datapull.sas) is a SAS macro that you can invoke to download the data files as you need them. You should note that both programs will only work in the SAS environment. You will learn more about the code for both programs in the first two chapters of the book.

SAS OnDemand for Academics

This book is compatible with SAS OnDemand for Academics. If you are using SAS OnDemand for Academics, then begin here: https://www.sas.com/en_us/software/on-demand-for-academics.html.

Where Are the Exercise Solutions?

One of the great features of this book is the practice exercises at the end of each chapter. They will enable you to hone your skills in some of the techniques that you will come across in the book. You can request the solutions to the practice exercises through the book's GitHub repository.

About The Author

Babatunde Odusami, PhD, CFA, is a professor of finance at Widener University in Chester, PA. Alongside his academic roles, he's engaged in investment management and governance, corporate governance, and consulting. He received an MBA with a concentration in management information systems, as well as an MS and PhD in financial economics from the University of New Orleans. He is also a CFA charterholder. His research interests are in statistical and machine learning models that can explain how the values and risks of financial assets and portfolios evolve. His commentaries on financial topics are periodically featured in print and TV media.

Learn more about this author by visiting his author page at https://support.sas.com/en/books/authors/tunde-odusami.html. There you can download a free book excerpt, read the latest reviews, get updates, and more.

Chapter 1: Financial Data Science: An Overview

Introduction

The primary job of a data scientist is to create value out of data. Although organizational data is commonly accepted as an intangible asset that presently cannot be capitalized on the balance sheet, it is now one the most lucrative means to create value. Indeed, the business models of some of the most valuable companies today are built on monetizing the data that they collect from the end-users of their platforms. As these types of companies continue to emerge and as other businesses continue to seek opportunities to extract value from their own data or data acquired from other businesses, the demand for data scientists is expected to continue to grow into the future.

There are academic programs such as computer science, data science, mathematics, and statistics that have a formal curriculum for those interested in pursuing a career in data science. There are also many data scientists who are self-taught as well as domain experts who pick up the skills along the way to enhance their functional knowledge of their business domain. Therefore, it is quite possible that we might see a reversal in the future where the data scientists are the domain experts working alongside a team of citizen data scientists who use data science techniques regularly, although their primary job function is not data science. This textbook is written for such professionals. It aims to provide professionals and students in graduate and advanced undergraduate programs with a rigorous foundation on the different types of data science tools in use in the financial services industry. The book is based on the SAS Analytics Platform, which provides end-to-end solutions for data science applications across all disciplines.

In this chapter, we will conduct an overview of financial data science. We will discuss the advantages and disadvantages of some of the common data science platforms that are currently available. We will also highlight some of the features of SAS that set it apart as the preferred tool for financial data science as well as showcase some simple practical applications of SAS in financial settings. We will conclude the chapter by delving into some of the key aspects of financial data science and emerging areas of concern as societies further embrace this novel approach to solving problems.

Data Science and Financial Systems

Data science and information systems are related fields in the sense that they both work with data. However, the focus of their relationships with data is distinct. Along the same lines, financial data science and financial information systems are also interrelated because they both work with financial data. Let's attempt to outline the differences between these domains in the next subsections.

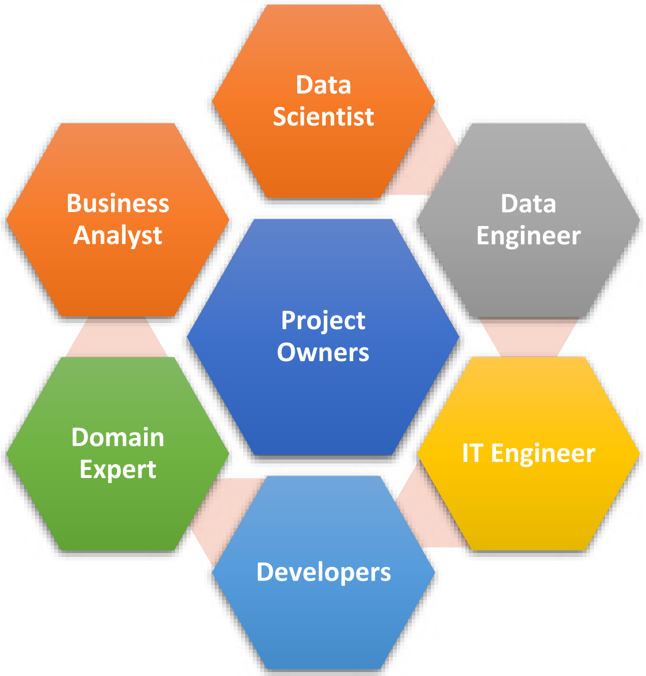
Data Science

Science is the study of events, structures, patterns, and phenomena in the real world, through observations, experiments, and testable justification and predictions about how these events and patterns occur. For example, natural sciences (such as physics, chemistry, and biology) study the natural world, while social sciences (such as economics, finance, sociology, and anthropology) study human societies and the social interactions that occur within them. Data science is a unique field of science in the sense that it also studies aspects of the real world but uses data as the primary artifact. Data scientists do not necessarily require physical observations or experiments when conducting research but rely mostly on advanced computational methods, programming, and statistics to extract insights about the real world from the data. Years of advancements in computing and Internet connectivity have led to a deluge of data. Indeed, data is arguably the most ubiquitous resources available today. However, in the same way we create value by turning raw materials into finished products through our manufacturing processes, data also needs to be processed and analyzed to extract value from it. This requirement has led to the development of advanced analytical tools to sift through these massive amounts of data for actionable knowledge that might be buried in them. These advanced analytical tools include those that are created to allow us to depict large amounts of data in visually compelling forms, as well as those that allow us to iteratively combine, transform, explore, model, simulate, and discover hidden patterns in the data.

Another unique attribute of data science is its interdisciplinary applications. Data science techniques and data scientists can be found across all other disciplines. Indeed, most data scientists do not require extensive domain expertise to work in cross-disciplinary teams, which will typically include domain experts who will possess a deeper understanding of the essential body of knowledge in the areas of inquiry. Therefore, data scientists are unencumbered in terms of the business disciplines where they can practice their craft. For example, a data scientist could be working with a team of doctors to understand how patients are responding to specific types of therapies or with the marketing team of a firm to understand what types of sales incentives are more likely to elicit desired purchase decisions from current or prospective clients. Readers are more likely to see that the same data science techniques commonly used in one business discipline often transcend that discipline. This is because regardless of the area of inquiry, the underpinning of the tasks in which data science techniques are used is the data itself.

Shown in Figure 1.1 is a typical configuration of data science teams. In financial settings, the project owner could be a portfolio manager or a risk manager. Project owners typically bear the

Figure 1.1: Data Science Team Configuration



ultimate responsibility for the success or failure of the project. Data engineers are responsible for maintaining the software components of the infrastructure that are used for collecting, processing, and retrieving the business data that will be analyzed and modeled during the project. The main responsibilities of IT engineers are to design, install, and maintain the hardware component of the data infrastructure and any other systems that work in tandem with the data infrastructure to ensure that the project runs smoothly. The developers typically work with the data scientist to design and produce business applications and dashboards that incorporate the models developed by the data scientist as their building blocks. A domain expert would be someone with deep knowledge of the theory and/or practice in the area of inquiry such as a financial economist, trader, or risk officer. Business analysts are members of the team that is responsible for generating data-driven analysis for decision-makers. In finance settings, these would include investment or credit analysts. The data scientist interfaces with all of these entities to develop the algorithms and models that would solve the business problem that required the project.

Financial Data Science

Financial data science is an emerging sub-field of data science. It involves the creation of ad hoc analysis to answer specific business questions and forecast possible future financial scenarios. Finance as a sub-field of economics often uses theories to explain or predict the relationships between financial and economic variables.

Most financial theories do not work very well in the real world because they rely on assumptions that are often unrealistic or theories based on abstractions of the real world that are too limiting to function in modern societies. For example, a well-known financial theory asserts that the prices of financial assets evolve in a random manner such that it is impossible to accurately forecast their future prices by simply observing past price changes. The motivation for this theory is that rational, profit-maximizing investors will quickly arbitrage away any such predictable patterns. Thus, the only patterns that are observable in the financial markets are the unpredictable ones. This theory implies that any attempt to predict the future directions or specific attributes of financial markets using statistical or algorithmic models is essentially a futile effort. However, the conjecture of this theory is inconsistent with the reality that between 60-73% of equity trades and large proportions of fixed-income and currency trades that are executed in each of these markets are performed by applications that were developed using data science tools. It may be that these patterns are indeed predictable and that most investors lack the means or motivation to exploit them, leaving only those with the motivation and computational resources to do so. Even if they are not predictable as postulated by the theory, data science still provides many tangible benefits to the organizations that deploy them, as you will see shortly.

While theories might not provide perfect insights into how financial data behave, they still provide a solid framework to bring to bear other tools and methodologies that can further our understanding of how financial and economic variables behave in the real world. Thus, financial data science involves the application of new and established data science methods and techniques to business and financial data, to gain new insights into trends and relationships in the data that are not previously revealed in standard financial theories and models. It is important to note that financial data science is not a substitute for conventional financial and economic theories. Data science and theory are merely complementary tools that allow us to extend the scope of our understanding of how financial variables behave and consequently derive better forecasts of their future directions.

Business Applications of Financial Data Science

Finance is a business discipline that is mainly concerned with how financial resources are raised, allocated, and used to achieve an increase in the stock of wealth of individuals or organizations. There are some elements of risk entailed in each of these activities; hence, finance is also involved in risk identification, analysis, mitigation, and governance. Financial activities often generate large amounts of data, which generally have been collected and archived in a well-structured manner. For example, at the macro-level, there are publicly available and proprietary databases containing daily stock market information on all publicly listed companies in the US since the 1900s. As well as data on various economic variables, which have been systematically collected and curated for over 100 years. Organizations also possess large amounts of financial data that they typically collect for operational needs, such as revenue and cost data, which are normally collected for financial reporting or financial planning purposes. Hence, financial data are well-primed for the application of data science techniques.

Arguably, the financial services industry is one of the first industries to deploy data science techniques as tools for day-to-day business processes. These include a wide range of applications in the field of investment and risk management, as well as financial planning and forecasting. In investment settings, financial data science techniques such as predictive modeling, simulations, and optimizations are broadly used for decision-making, asset allocation, trading strategies, risk, and performance measurement purposes. In corporate finance settings, simulations and optimization are also used for capital sourcing, capital expenditure analyses, revenue and profit optimization, and risk management. In insurance and financial intermediation settings, predictive modeling, simulations, and optimizations are used for risk measurement, analysis, pricing, and governance.

More recent innovations in the applications of financial data science techniques are in the financial technology (FinTech) spaces, where all of the previously mentioned tools are used to create platforms and products that are essentially designed to disintermediate the flow of funds between lenders and borrowers, and investors and investment opportunities. In subsequent chapters of this book, we will explore various data science techniques and their common applications in finance settings.

Financial Information Systems

Information systems are organized methods for collecting, processing, archiving, reproducing, and analyzing business data. Financial information systems are perhaps the most widely used information systems since all organizations require some type of organized approach to managing their financial records and making plans for their future financial needs and investments. Modern financial information systems, which could range from a simple Microsoft Excel workbook or Access database to mid-level accounting software such as Intuit QuickBooks and Oracle NetSuite, or even more sophisticated enterprise or custom solutions, are typically computerized systems that collect, process, archive, reproduce, and provide analysis of the financial data that are stored in them.

Financial information comes in different forms depending on the needs of the entity. For a retail organization, financial information could be sales data collected through interfaces such as point-of-sales (POS) systems, or price data received as Extensible Business Reporting Language (XBRL) file from the supplier, or inventory data, which are created and tracked using radio frequency identification (RFID) tags. This information is then processed to fit a specific format and archived in the databases, for future reproduction and analysis depending on the organization's needs.

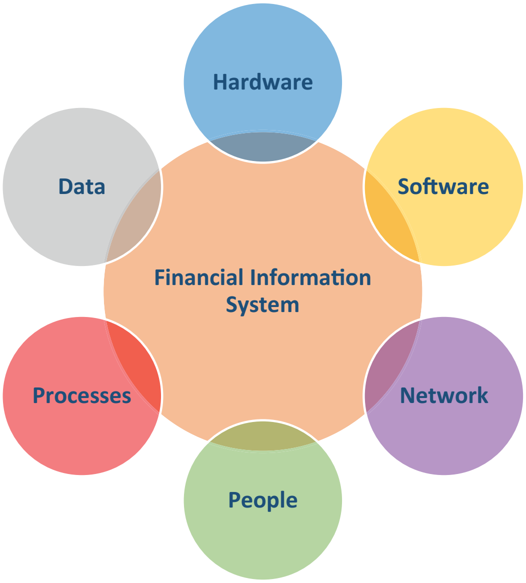
Financial information systems used in the financial services industry vary from those used in retail and manufacturing organizations because of the unique nature of the industry. The industry is highly regulated, and its operations and financial structure are usually different from other business sectors. The financial services industry is also quite broad. It is comprised of a wide range of businesses, including depository and non-depositary institutions, investment companies and intermediaries, capital market makers, insurance companies, and the more recent financial technology (FinTech) companies.

Components of Information Systems

All information systems consist of elements that work together to meet the organizational objective. In general, components of financial information systems include:

- **Hardware**: This is the physical component of the information system, including computers, peripherals, and media devices. Peripheral devices include input (such as POS, barcode, and QR code readers), output devices (such as displays, printers, and speakers), and media devices, which are disks on which the information is typically stored.
- **Software**: Two types of software are used in information systems. System software is the program used to control the hardware. Application software includes sets of packaged code that are used to collect, archive, reproduce, and analyze information. System software provides the interface between the hardware and application software, while application software executes sets of tasks that are typically unrelated to the operation of the computer itself. Many readers are familiar with Microsoft Windows, which is the most popular family of operating system software in the market. The SAS software that will be introduced in more detail in subsequent sections of this book is an example of an application software that can be used for collecting, processing, archiving, reproducing, and conducting statistical analysis of the data.
- **Network**: Information systems require network communications to effectively work because their data repository often needs to be accessed by multiple users, from multiple locations, and using different devices. Hence, information systems must have a network architecture (wired or wireless connections and network topology) for the devices, system, and application software to communicate with each other. Networks could be built to allow access only within the organization (intranet) or beyond the organization (extranet) and, more recently, in the cloud, in which case all or some of the IT infrastructure of the organization is hosted in public or private platforms hosted by other organizations.
- **People**: All organizations require people to operate, and since information systems are designed for organizational use, people are critical elements of all information systems. These include end-users who use the information system to carry out their respective business tasks, developers who build applications and technologies that run the systems, as well as administrators and system specialists who ensure the systems operate as intended. Examples of end-users would be a payroll specialist who uses the human resource database to run payroll reports every month, a financial data scientist who uses the loan portfolio database to build credit scoring models for the bank, and developers who build applications such as dashboards, graphical user interfaces, program packages, and manuals that allow other users to access resources in the system.
- **Processes**: These are the methods used in the governance and operations of information systems to ensure that they achieve their intended design. For example, processes could include tasks and procedures relating to how data is collected, organized, stored, altered, retrieved, and transmitted within the system (data processing). They would also include the access level available to users and devices within the systems (controls) and procedures that are manual and those that are automatically executed in the system.

Figure 1.2: Components of Financial Information Systems



- **Data and Databases**: It is straightforward to think of data as pieces of relevant information that are collected and stored in data repositories that are popularly called databases. Within this context, financial data can then be regarded as any piece of data that has financial relevance to an entity. Financial data could exist either as structured or unstructured data in its raw form. Structured financial data are pieces of information that have been collected and archived using predefined formatting, while unstructured data are typically archived without preformatting. We will discuss these two concepts in more detail in Chapter Two. Financial data scientists mostly work with structured financial data. The three types of data structures that are commonly used for archiving structured financial data include:
  - **Cross-sectional Data** – Data on a statistical unit or items of interest that are collected at a single point in time. For example, the company names and the industry of the stocks in a portfolio at the end of the quarter.
  - **Time Series Data** – Data on the same item or statistical units that are collected over multiple periods. For example, the daily closing price of a stock in the portfolio that was collected over two years.
  - **Panel (Longitudinal) Data** – Cross-sectional data that are collected over multiple periods. For example, the daily closing prices of all the stocks in the portfolio that were collected over two years.

Regardless of the data structure or the format that you will encounter while implementing your analytics project, note that you will still end up spending a significant amount of your time preparing your data for modeling.

Financial Intelligence

Along the same line as business intelligence, which leverages the power of software and databases to draw insight from past events and outcomes, financial intelligence applies similar tools to historical financial data to draw insights about relevant key performance indicators (KPIs), scorecards, dashboards, or any other metrics that are of value to the decision maker. For example, portfolio managers use portfolio reporting tools to visualize and analyze the performances of their investment portfolios. Banks use loan dashboards to monitor and analyze applications, approvals, payments, and default trends in their loan portfolios. The primary difference between financial intelligence and financial analytics is the window of opportunity that is under consideration. With financial intelligence, the emphasis is on figuring out what happened in the past so that decision-makers can judge how well the organization or portfolio is meeting its intended objectives. Whereas in financial analytics, the emphasis is on figuring out what will happen in the future so that decision-makers can exploit those insights for business purposes.

Financial Econometrics

Financial econometrics explores financial and economic problems and theories by applying inferential statistics, as well as structural, and descriptive models, to financial and or economic data. In financial econometrics, the theory is normally the starting point, followed by the implementation of the model to prove or disprove the theory. Econometric models are abstractions of the real world that are designed to test theories, underlying assumptions, or forecast future trends. In one way, financial econometrics can be thought of as a subset of financial data science that sets out to prove or disprove financial phenomena or relationships, while financial analytics are designed to find financial phenomena and relationships. With that said, it is important to reiterate that financial and economic data often occur in time series format, which raises a range of issues that many of the advanced data science tools are not necessarily equipped to tackle. For example, financial and economic data often display a wide range of statistical characteristics such as time-varying volatilities, trends, cyclicality, seasonality, outliers, serial correlations, and endogeneity to name a few. Thus, financial data scientists must pay special attention when applying advanced data science tools such as machine learning to financial and economic data. In subsequent chapters, we will discuss in more detail how to address some of these features in the data science framework.

Data Science Toolkit

One of the privileges of being a data scientist today is the wide array of tools at our disposal. Data science tools fall into two categories: those based on open-source platforms such as Python and R and those based on proprietary platforms such as Microsoft Excel, SAS, SPSS, Tableau, and MATLAB. Each of the platforms has its benefits and disadvantages, which are discussed in the succeeding section.

Microsoft Excel

Microsoft Excel is arguably the most widely used data science application. Since its introduction in 1987, Excel has grown to become the leading spreadsheet and perhaps the easiest to learn data science application. To an average user, Excel might look like a simple spreadsheet that contains data items that are organized into rows and columns. However, behind the scenes is a wide array of powerful analytic and data science engines that users with minimal computing skills can implement for data science purposes. Indeed, many of the tasks that would require advanced Excel skills in the past now have menu options or have been automated such that minimal programming is needed to implement. For example, users can enable the Analysis ToolPak and Solver Add-in to access the statistical data analysis and optimization and equation-solving solutions in Excel.

More recent versions of Microsoft Excel also include data engines to connect to most databases, data lakes, and data files, as well as web scraping tools that can pull data directly from online data sources. It also includes powerful visualization tools and data query tools that run on an artificial intelligence (AI) platform. Together, these features allow users to access needed data relatively quickly, conduct drill-downs and basic data analyses, and produce compelling visualizations with minimal computing skills. Microsoft Excel format is also the most common format in which data is stored and accessed by the other and more sophisticated data science tools that will be discussed shortly.

IBM SPSS Statistics

Since its acquisition by IBM in 2009, IBM SPSS statistics has grown to be a major contender in the data science landscape. Despite its roots in social science research, SPSS is equipped with multiple advanced features that enable it to support the needs of both novice and seasoned data scientists. Most users will find its graphical user interface easy to navigate. Users can also execute simple and complex analytics tasks and produce visualizations using custom-built menus.

SPSS supports the use of structured query language (SQL), which allows it to connect directly to databases. It also supports data in multiple file formats such as Excel, CSV, SAS, and Stata. For more advanced users, SPSS supports three types of programming languages: its own native SPSS syntax, as well as the R and Python programming languages, which we will discuss shortly. Users with an interest in implementing more advanced and automated data science techniques can also subscribe to the IBM SPSS modeler.

The SPSS platform has some key disadvantages when compared to other data science platforms. First, SPSS does not have a robust visualization engine, so output graphics tend to be of lower quality than those produced by other platforms. SPSS was not initially designed to handle financial data, which tends to follow a time series format, so financial data scientists will find it limiting in terms of the number of prebuilt menus and functions that can analyze financial data and support the writing of programs for advanced financial data analysis. SPSS also lacks a visual programming platform that can be used to manage and automate tasks, routines, and subroutines on data science projects. The last disadvantage of SPSS is cost. Students and faculty interested in using SPSS must pay for an annual license.

Tableau

Visual exploration of data is a crucial aspect of data science. Indeed, visualizations allow data scientists to quickly observe and communicate trends, intensities, and relationships in large amounts of data. Tableau is perhaps the most widely used data visualization application. It is easy to learn, and users can quickly produce graphically compelling and interactive visualizations without advanced programming knowledge. Tableau can also automatically pre-process and post-process very large amounts of data quickly and link data in different formats together. Tableau probably has the most comprehensive list of data connectors (over 90), which allows it to connect to data stored in various file formats as well as those in open-source databases such as MySQL and PostgreSQL. It also has a webscraping tool that can pull data directly from online data sources such as Google Analytics. With some configurations, Python can be integrated into Tableau to access some of the advanced data science features that are not native to Tableau but are readily available in Python. Together, all these features allow Tableau users to quickly draw insights from data and report their findings using high-quality graphics.

However, it is important to note that Tableau, at its core, is a business intelligence application that is well-suited for reporting purposes but not for conducting advanced data science techniques, such as machine learning and deep learning. Some of the advanced features of Tableau, such as access to data on servers and integration with Python, require significantly higher levels of expertise in computing, which makes it challenging for novice users. Finally, there is also an annual cost for its license and some users might find that prohibitive.

MATLAB

Those coming to data science from the engineering and science disciplines might already be familiar with MATLAB due to its popularity in the engineering and scientific fields. It is a proprietary programming language for technical computing and modeling in the scientific fields. With a wide range of built-in functions and routines, and a robust ability to manipulate and visualize data stored in matrix format, MATLAB can be a potent tool for data science. Indeed, one of the advantages of MATLAB is the growing number of toolboxes that it has for data science applications. For example, MATLAB currently has a Statistics and Machine Learning Toolbox, Deep Learning Toolbox, and a Text Analytics Toolbox, to name a few. These toolboxes, along with its long-available Econometric, Financial, Math, and Optimization toolboxes make MATLAB a comprehensive arsenal for advanced financial data science. However, readers interested in using MATLAB and its associated toolboxes for data science might find the cost to be quite prohibitive for use as a learner. Consequently, MATLAB has a smaller ecosystem and community of users, relative to the other data science platforms.

Python

Python is a high-level and scalable open-source programming language with a wide range of applications. It is concise, easy to read, and supports an object-oriented programming approach. It is also an interpreted language, so it does not need a compiler to run. It is used for technical computing, data science, web development, and application programming. Therefore, Python also has a large user community that spans multiple professions. It is able to achieve such versatility because Python supports an extensive list of libraries that contain modules and/or packages.

Python modules and packages are collections of reusable Python code that perform related tasks. Python programmers can quickly call up these code in other programs without the need to rewrite them all over again. For example, a developer who is conducting numerical analysis in Python can call up the NumPy (Numerical Python) library within a Python program to execute mathematical operations such as linear algebra, Fourier transformation, matrix analysis, and random simulations. There are other Python libraries such as Pandas, which is used extensively for data processing and analysis; Matplotlib, which is used for data visualization; SciPy, which is used for technical computing tasks such as optimization, integration, and signal processing; and Scikit-learn, which is used for advanced data science tasks such as predictive modeling and machine learning. Python also supports cross-platform integrations. Indeed, many of the current proprietary data science applications such as SAS, SPSS, and Tableau integrate with Python. Thus, users can switch back and forth between Python and proprietary applications and essentially get the best of both worlds.

All these features make Python well-suited for data science applications. Within the data science community, it is arguably the most widely used data science application. Despite its impressive list of features, Python has some limitations that novice users might find challenging to overcome in their data science journey. Python does not have native support for data connectors to enterprise data repositories. It is also memory intensive and slower than other high-level languages such as C++. Nevertheless, it is highly recommended that aspiring data scientists acquire some functional knowledge of Python programming, irrespective of their preferred platform.

R

In contrast to the versatility of Python, R is an open-source statistical programming language that has a variety of data science functionalities. It also has a large community of users, but they are mostly in the academic and research space. It shares some similarities with Python in the sense that it is an interpreted language and supports an extensive list of R packages. Indeed, there are over 19,000 packages that have been published for R users. These include packages for executing a wide range of statistical analyses, data visualizations, advanced econometric models, mathematical operations, and optimizations, as well as packages for advanced financial data science tasks such as predictive modeling and machine learning. R can also integrate with proprietary applications such as Tableau, SAS, and SPSS. Many of the advanced data science functions and routines in some of the proprietary applications are essentially wrappers around R packages running in the background.

R can also be installed as a standalone application, in which case the user will need to rely on codes to interact with the application, or use RStudio, which adds a graphical user interface with menu functions and a syntax editor to R. Packages in R often lack the transparency and comprehensive support resource that are much easier to access for the libraries and functions of similar data science platforms.

SAS

The SAS analytic suite is possibly the best-suited platform for data science. It offers a comprehensive suite of data science tools that span every aspect of the analytic and business intelligence life cycle that a data scientist can possibly encounter. As with most data science applications, SAS is at its core a statistical programming language with a wide range of applications that transcend all business domains.

There are several appealing features of SAS for aspiring and seasoned data scientists. First, it is a versatile and powerful programming language that is easy to learn. All SAS users appreciate its robust support infrastructure, which is built on a vast repository of SAS documents, sample code, technical support, training programs, conferences, and a passionate user community. There is also a broad range of tools available to users at various levels of SAS expertise. Beginner and advanced users will find many of the menu-driven SAS applications (which still retain their programming capabilities) such as Enterprise Guide and Enterprise Miner particularly useful in their analytics journey. Others will find the flexibility and on-demand access to the SAS engine through web-based platforms such as SAS Studio and SAS Viya extremely convenient. Besides these, SAS also has a comprehensive list of data connectors that allow it to connect to data stored in various file formats, including data in open-source and proprietary data repositories, as well as powerful reporting tools that can automate the analytic life cycle for most data science projects.

SAS has robust capabilities for advanced financial data science applications in artificial intelligence and its subfields such as machine learning, deep learning, computer vision, natural language processing, and financial econometrics. It integrates seamlessly with Python and R, such that users can combine SAS code with these programs in the same analytics environment. Although it is a proprietary solution, SAS offers free software options for learners in both academic and non-academic communities through its SAS OnDemand Platform.[1]

Another appealing feature of SAS is its credentialing program. SAS users can demonstrate their competence in SAS by enrolling and passing one or more of the certification exams offered by SAS. SAS is also a market leader in the analytic space, and the demand for SAS talent remains very strong. Lastly, from a risk management point of view, users can be sure that all SAS products

[1] To access SAS Studio for free on the web, readers should go to https://welcome.oda.sas.com/ to create a free SAS profile. This will provide you access to SAS Studio and 5GB of free storage for your personal data files.

and procedures have been subjected to rigorous testing before their release, and there is a single point of accountability for future upgrades, a feature that is lacking in many of the open-source platforms. All of these features make SAS a compelling tool for financial data scientists and the primary application that will be highlighted in this textbook.

Working with SAS

Although the book does not assume that readers have significant SAS programming skills, many of the concepts we will discuss in succeeding sections of the text do require some foundation in finance, mathematics, statistics, and computer information systems. Hence, one of the aims of the book is to provide these readers with advanced knowledge of how these fields are interrelated in the financial services industry. Users interested in working with SAS will be delighted by the assortment of environments through which they can access the SAS analytic engine. In enterprise settings, the SAS engine (the current version of which is SAS 9.4) is usually located on a SAS server that can be accessed by client applications. On personal computers, the server is locally installed and can be accessed using the SAS Windowing Environments (Explorer, Results, Enhanced Editor, Log, and Output windows), SAS Enterprise Guide, and SAS Studio.

It is also important for you to be aware of SAS Viya, which is the newest member of the SAS Analytics Platform. SAS Viya is a full suite of cloud-based applications with artificial intelligence, data visualization, advanced analytics, and data management features that allow it to support the entire analytic life cycle. Although it shares some similarities and interoperability with SAS 9, it was built from the ground up to support processing in-memory and distributed processing. It also has its own programming language, known as the cloud analytics services (CAS) language. However, it supports the SAS programming language.

Windows in the SAS Windowing Environment

PC users can also access SAS using a powerful but menu-based desktop application such as the SAS Enterprise Guide, or web-based client applications such as SAS Studio. Advanced analytic applications such as SAS Enterprise Miner are used throughout the entire scope of the data science project. In this textbook, we will focus on three SAS environments: SAS Enterprise Guide, SAS Studio, and SAS Enterprise Miner. There are some similarities between the three environments. All three are menu-based but also have robust programming interfaces, such that users can seamlessly switch back and forth between point-and-click menu-based tasks and writing code to implement unique tasks. All three environments also support automation for repetitive tasks, as well as provide a mechanism to organize a sequence of tasks (process flow), data items, and results into a single repository called projects. Each menu-based task is usually a packaged set of code, which all three windows generate as the menu-based task are implemented. Novice users will also find these features to be very helpful for writing future code or customizing the software-generated code for their own unique tasks.

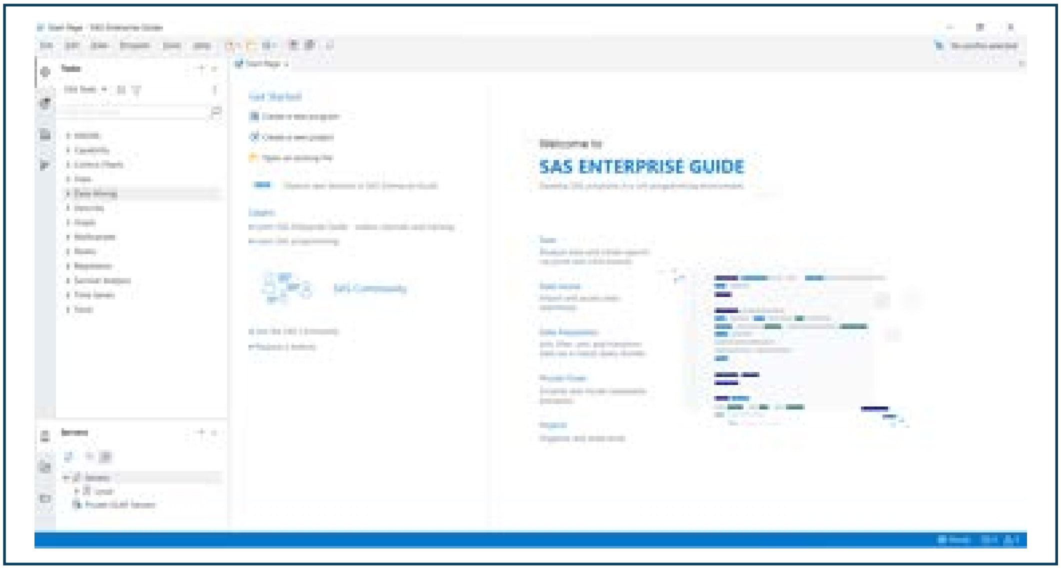
Many readers would be delighted to learn that financial data science in SAS does not always entail writing SAS programs. For many tasks, it might be more efficient to use menus than to write programs to implement them. Nevertheless, all data scientists must be highly competent in programming and be ready to apply their programming skills when there are no menu options to implement a task.

SAS Enterprise Guide

SAS Enterprise Guide is a point-and-click desktop client for working with SAS and managing analytic projects. As you click on the task menu, the SAS Enterprise Guide generates SAS code behind the scenes. The SAS code is then submitted to a local or remote SAS server for processing. Enterprise Guide also has a full programming interface that can be used to write, edit, and submit SAS programs to a SAS server for processing. The software also has other project management features such as the process flow tab, which allows you to manage and track your analytics project from end to end. You can also automate and schedule the execution of your completed project as well as share any elements of your project using process flow.

Enterprise Guide 8.3 is fully integrated with GitHub, a platform for collaborating and tracking changes on software development projects. Users can also connect to the SAS Viya platform using the SAS Enterprise Guide.

Figure 1.3: SAS Enterprise Guide 8.3 Environment



SAS Studio

SAS Studio is a web-based interface for working with SAS. In SAS Studio, SAS programs are sent to a local or cloud-based SAS server using common web browsers. The server processes the code and publishes the output in various formats, including HTML, RTF, and PDF. SAS Studio also supports a comprehensive list of point-and-click menu tasks, which can be used to implement both basic and advanced analytics procedures.

The cloud-based version of SAS Studio provides access to the SAS engine from anywhere with an Internet connection. SAS Studio also shares many of the features available in SAS Enterprise Guide, such as process flow and connection to the cloud-based SAS Viya. Although it runs on a browser, SAS Studio can be installed as a Progressive Web App (PWA). This approach provides more user-friendly features, such as placing an icon for SAS Studio on the desktop of your computer. This means you can skip multiple steps to reach the SAS environment because the steps are automatically performed once you click the SAS Studio icon. PWA also enables application persistence, which allows the user to remain logged in to the server unless the time-out feature is enabled. You can also create multiple icons for each instance of PWA.

SAS Enterprise Miner

SAS Enterprise Miner is another point-and-click SAS application that is used for building descriptive and predictive models of large data. The software supports a wide range of data management, statistical procedures, and analytics algorithms, all of which can be accessed by simple point-and-click

Figure 1.4: SAS Studio Environment

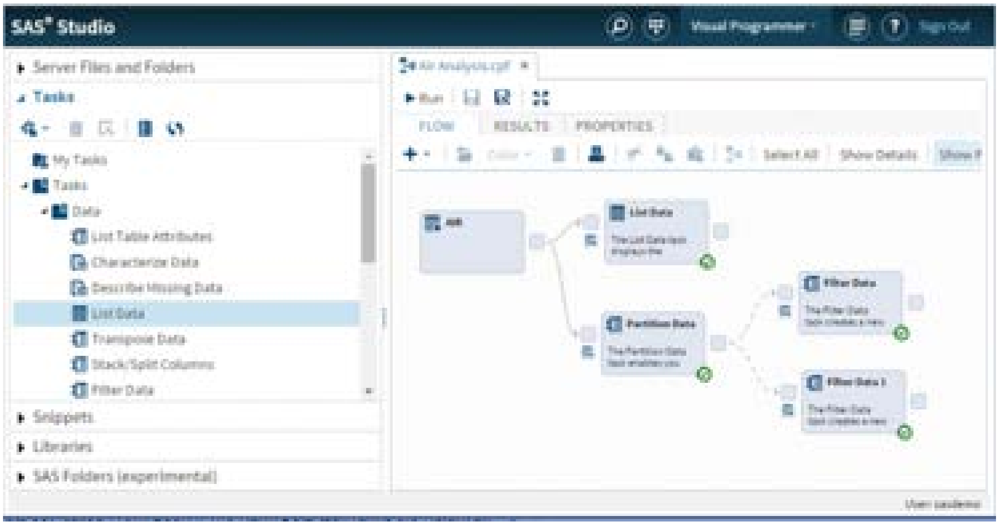
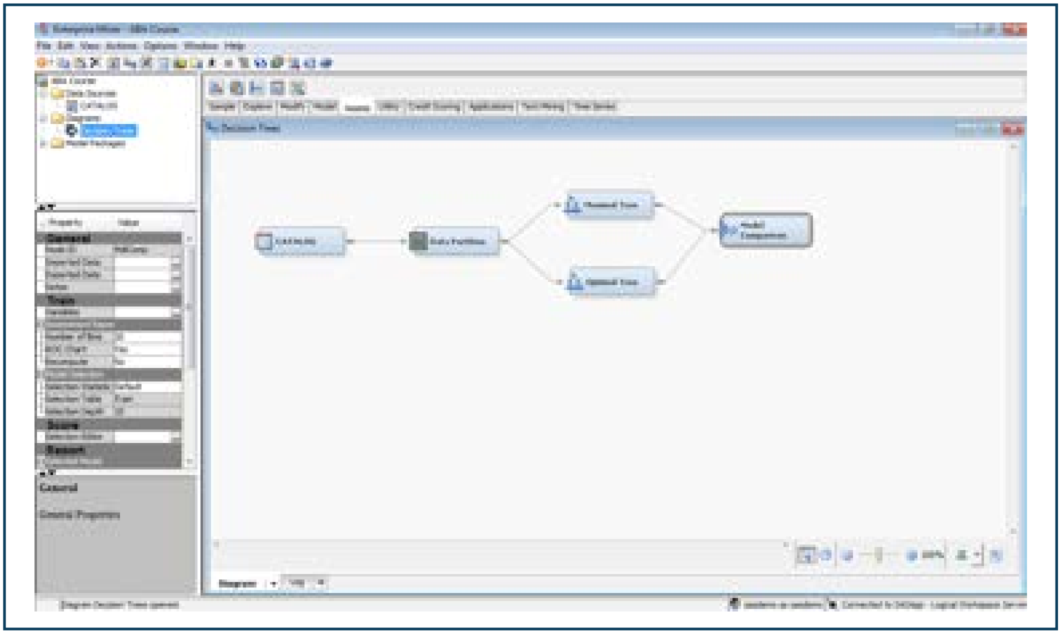


Figure 1.5: SAS Enterprise Miner 15.2 Environment

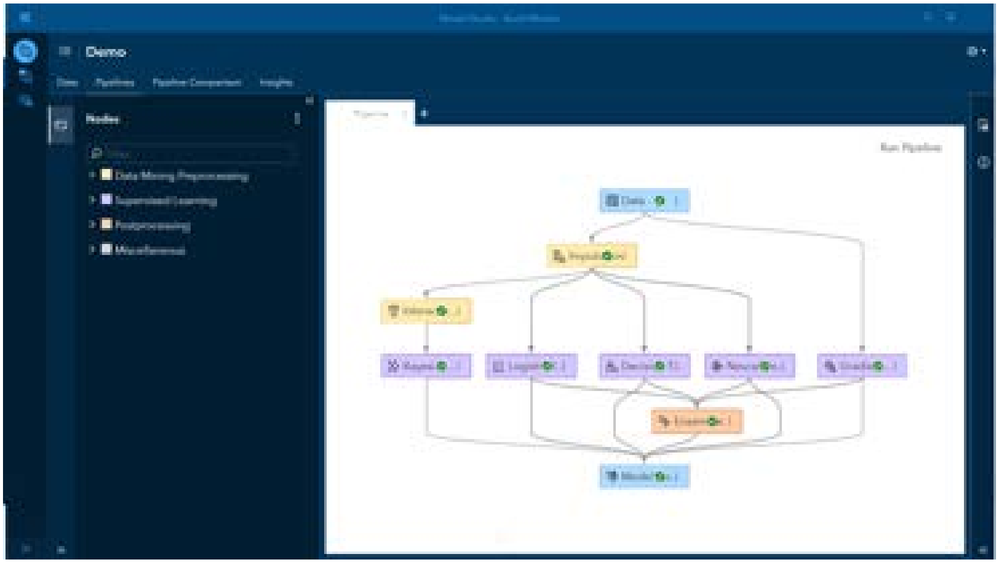


actions. Most of your analytics tasks in Enterprise Miner will be done in the process flow diagram using pre-built code packages, which are called Nodes. However, the application still supports full SAS programming capabilities as well as the ability to deploy analytics models into production within the software environment. Enterprise Miner projects can be imported into SAS Viya. Another great feature of Enterprise Miner is its integration with R and Python. With some programming, R packages and Python modules can be integrated into the Enterprise Miner process flow.

SAS Model Studio

Although you can access SAS Viya through any of the three previous platforms if you have a license to the SAS/CONNECT bridge, most users will find it more beneficial to use SAS Model Studio as the default application because it is native to the SAS Viya platform. SAS Model Studio is an integrated visual environment that provides access to a suite of analytics products and features that are built on the SAS Viya platform. The list includes data management and governance, visual data mining and machine learning, visual text analytics, visual forecasting, visual model management, optimization, and robust support for the integration of open-source platforms such as Python and R. You can also execute code written in both the SAS and CAS programming languages in SAS Model Studio. Pipelines are a key feature that SAS Model Studio shares with the previous windowing environment (pipelines are what process flows are called in SAS Model Studio). Just like SAS Studio, SAS Model Studio can also be installed as a PWA.

Figure 1.6: SAS Model Studio



SAS Statements

SAS programming covers a wide range of steps, procedures, and functions. Unfortunately, not all can be discussed in this book. Hence, we will focus only on the code and functions that are most relevant for financial data science purposes.[2] All code written in the SAS programming language can be grouped into two broad categories: the DATA step and PROC statements (also known as SAS procedures).

DATA Step

The DATA step is a group of SAS statements that are used for importing and manipulating data in SAS. It usually begins with DATA as the initial statement, followed by blocks of code that SAS sequentially executes. The DATA step normally ends with a RUN statement. All data, regardless of their current format must first be read and stored in SAS before they can be accessed by other SAS statements. There are various ways to read your data into SAS, depending on the current format of the data. In the example below, we create a new SAS data set called SP500FIN by entering the data directly into SAS. The raw data contains the aggregate annual sales per share

[2] Those interested in developing a broader range of SAS competencies should visit the SAS bookstore at https://support.sas.com/en/books.html for other great books such as *The Little SAS Book: A Primer and Learning SAS By Example: A Programmer's Guide.*

(SPS), earnings per share (EPS), dividend payout (DPR) ratio, and price-to-earnings (PE) ratios for all companies listed in the S&P 500 index from 2015 to 2022.

Program 1.1: Reading Raw Data into SAS

```
data SP500FIN;
      input  Date MMDDYY10. SPS EPS DPR PE;
      format Date MMDDYY10. SPS Dollar10. EPS Dollar10.;
      label SPS = 'Sales Per Share' EPS ='Earnings Per Share' DPR= 'Dividend Payout
Ratio' PE = 'Price-to-Earnings Ratio';
      datalines;
12/31/2015 1106.96 89.73 53.38 18.73
12/30/2016 1128.45 98.90 52.35 20.44
12/29/2017 1210.13 109.99 52.11 21.48
12/31/2018 1313.58 133.01 46.10 16.64
12/31/2019 1391.09 140.42 52.45 20.86
12/31/2020 1342.44 97.00 72.10 30.37
12/31/2021 1541.77 200.35 37.18 24.71
12/30/2022 1708.91 188.41 39.34 18.61
;
run;
```

The DATA statement creates the SAS data set named SP500FIN. The INPUT statement assigns variable names to the columns. The MMDDYY10. Is an INFORMAT statement that tells SAS how to read or input data (in this case to read the date in MM/DD/YYYY format). The FORMAT statement tells SAS how to display the data. The LABEL statement assigns variable labels to the variable name and the DATALINES statement indicates the beginning of the observations of the values of each variable.

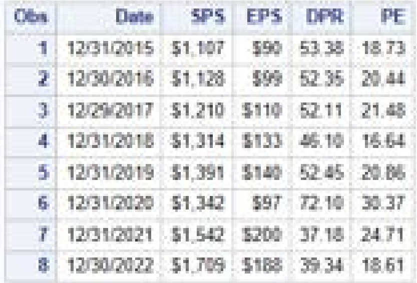
PROC Statements

The second group of SAS statements is SAS procedures or PROC statements. These are used to execute a variety of tasks in SAS. They include statistical analysis, econometrics, data management, visualizations, reporting, and advanced analytics to name a few. When implementing a PROC step in SAS, you generally need to refer to the data set on which the procedure will be executed. Hence, most PROC statements will include a “DATA=” in the code line as shown in the examples below. In the next code example, we sort the SP500FIN data set by date using the PROC SORT statement and then request a print of the sorted data using the PROC PRINT statement.

Program 1.2: Sorting Data by Date

```
proc sort data=SP500FIN;
by Date;
run;

proc print data=SP500FIN;
run;
```

Output 1.2: Printing SAS Data Set Sorted in Ascending Order



Obs	Date	SPS	EPS	DPR	PE
1	12/31/2015	$1,107	$90	53.38	18.73
2	12/30/2016	$1,128	$99	52.35	20.44
3	12/29/2017	$1,210	$110	52.11	21.48
4	12/31/2018	$1,314	$133	46.10	16.64
5	12/31/2019	$1,391	$140	52.45	20.86
6	12/31/2020	$1,342	$97	72.10	30.37
7	12/31/2021	$1,542	$200	37.18	24.71
8	12/30/2022	$1,709	$188	39.34	18.61

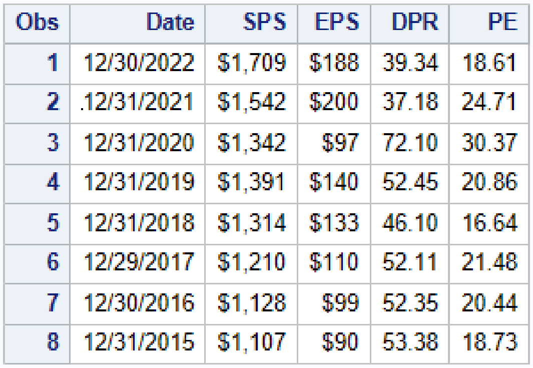
The default order for sorting in SAS is ascending, but you can change the order to descending. PROC SORT replaces the original data with the sorted data. However, you can also specify that the data should be sorted into a new data set by using the optional argument (OPTIONS) for PROC SORT.

Most PROC statements have optional arguments that specify how the procedure should be executed by SAS. Tasks in Enterprise Guide and SAS Studio have options in the built-in menus for each procedure. Enterprise Guide and SAS Studio also have a recommendation engine that suggests options for you as you write your code. You can learn more about the accompanying options for each statement by consulting the SAS documents for the procedure.

Program 1.3: Sorting Data by Date Descending

```
proc sort data=SP500FIN out=Dsorted_SP500FIN;
by descending Date;
run;

proc print data=Dsorted_SP500FIN;
run;
```

Output 1.3: Printing SAS Data Set Sorted in Descending Order



Obs	Date	SPS	EPS	DPR	PE
1	12/30/2022	$1,709	$188	39.34	18.61
2	.12/31/2021	$1,542	$200	37.18	24.71
3	12/31/2020	$1,342	$97	72.10	30.37
4	12/31/2019	$1,391	$140	52.45	20.86
5	12/31/2018	$1,314	$133	46.10	16.64
6	12/29/2017	$1,210	$110	52.11	21.48
7	12/30/2016	$1,128	$99	52.35	20.44
8	12/31/2015	$1,107	$90	53.38	18.73

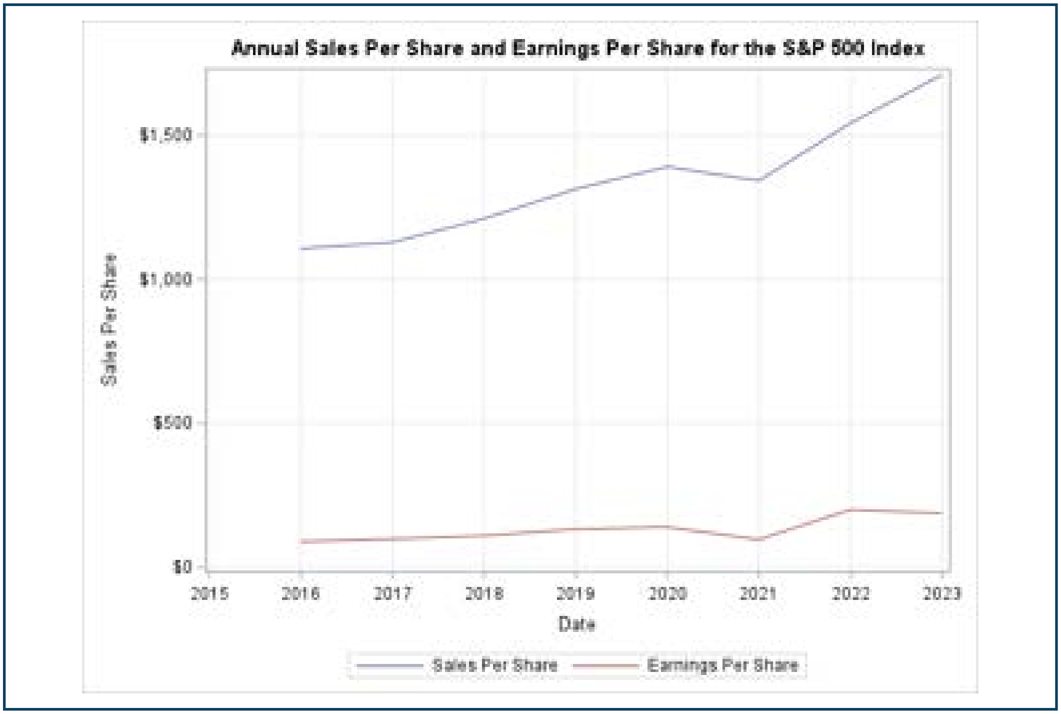
Output

Outputs from SAS DATA steps are usually new data sets created from data that is entered into SAS (as in the previous example), read from existing data sets, or imported from the data stored in many of the data file formats (such as Text, CSV, and XLSX) that SAS supports. Outputs obtained from PROC steps can take various forms. These include those in results form, which are tables and graphs that are published in various file formats, data, and reports, which are results that are compiled into document files. In the example shown in Program 1.4, we use PROC SGPLOT to create a plot of the annual sales per share (SPS) and earnings per share (EPS) for the S&P 500 index from 2015 to 2022. For each plot, we use the SERIES statement to specify the variables to plot on the X-axis (Date) and the Y-axis (SPS and EPS). The graph shown in Output 1.4 is the result you will obtain from running the SAS code.

Program 1.4: Series Plots of Aggregate Financial Performance of S&P 500 Firms Using PROC SGPLOT

```
title 'Annual Sales Per Share and Earnings Per Share for the S&P 500 Index';
proc sgplot data= SP500FIN;
      series x=Date y=SPS ;
      series x=Date y=EPS ;
      xaxis grid;
      yaxis grid;
run;
title;
```

Output 1.4: Series Plots of Aggregate Financial Performance of S&P 500 Firms



SAS Data and Library

SAS data are stored in SAS Libraries, which are collections of one or more SAS files that are recognized by SAS and can be referenced and stored as a unit in the local or cloud drive of the SAS server. Libraries are file addresses on computer drives that allow SAS to access files that SAS supports. There are two types of SAS libraries: permanent and temporary. Permanent libraries contain files that are permanently stored by SAS until deleted by the user. The files can be accessed in subsequent SAS sessions. Files in the temporary (WORK) library are only available during the current SAS session and are typically deleted once the session is ended. There are two types of permanent libraries, default SAS libraries and user-assigned libraries. Default SAS libraries are automatically created by SAS in each SAS session. They include SASDATA, SASUSER, SASHELP, and MAPS. User-assigned libraries are created using the LIBNAME statements. LIBREF is the SAS name for the library, followed by the physical address of the library on your computer drive between the quotation signs.

Although the data stored in the user-assigned library are permanent until deleted, the user will have to reassign the library in each SAS session to relink the physical address with the SAS library. Therefore, the LIBNAME statement and the accompanying LIBREF and physical address of the folder must be invoked in each SAS session to reassign the library.

> To create a SAS library, submit the following SAS command. Replace the area in italics with your preferred LIBREF and the physical address of the folder on your computer. LIBNAME *mylib 'c:\mysasdir\'*;

You can automate this process to ensure that your library persists across sessions by including an autoexec file in your Enterprise Guide project to automatically reassign your library every time you launch the project. For SAS Studio, use the GUI option to create your library. Right-click on My Libraries, include your LIBREF, and check Re-create this library at start-up.

Accessing the Data Repository for the Book

Most of the data and code used in this textbook have been made available on a GitHub repository (https://github.com/finsasdata/Bookdata). GitHub is a cloud-hosting platform for collaborative projects. SAS Enterprise Guide 8.3 and SAS Studio support full integration with GitHub.[3]

The data in the book's GitHub repository can be accessed in multiple ways. You can download the data and code into the preferred directory of your personal computer by visiting the GitHub repository for the book using a web browser. Users with SAS Enterprise Guide 8.3 and SAS Studio

[3] To learn more about SAS Enterprise Guide 8.3 Git integration, please review "Understanding Git Integration in SAS Enterprise Guide" in the *SAS® Enterprise Guide®* 8.3: User's Guide available at https://documentation.sas.com/doc/en/egug/8.3/titlepage.htm.

can also download all of the data and code into a SAS library named FINDATA by submitting the SAS statement in Program 1.5 below. To make it easy for readers to use the same code in both the Enterprise Guide and SAS Studio environments, all data files and programs that are pulled from the GitHub repository will be stored in the temporary SAS folder directory of your computer or the SAS OnDemand server.

Program 1.5: Access GitHub Data Repository Using SAS Git Integration

```
options dlcreatedir;
%let datapath = %sysfunc(getoption(WORK))/finsasdata;
libname findata "&datapath.";
run;

data _null_;
 rc = git_clone("https://github.com/finsasdata/Bookdata/",
   "&datapath.");
    %put rc=;
run;
```

It is also important to note that SAS Git integration can only clone the GitHub repository into an empty directory on your computer. You will get an error log if you try to copy the repository into a folder with existing files. If you encounter such an error, locate the physical address of the FINDATA library by submitting the SAS statement in Program 1.6 below, then delete or move all the files (including hidden files) into a separate folder.

Program 1.6: Identifying the Physical Address of SAS Libraries

```
proc datasets library=findata;
run;
quit;
```

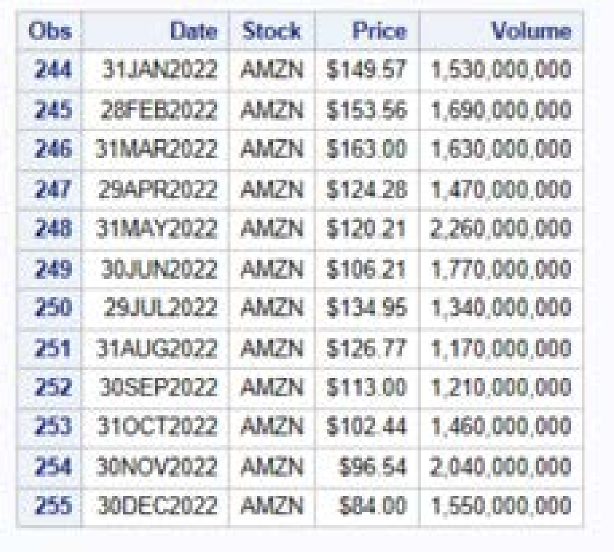
SAS can also retrieve files from the Internet by using HTTP requests. We can access one of the data sets in the GitHub repository by submitting the PROC HTTP statement in Program 1.7 below. The STOCKS data set contains the monthly stock prices and trading volume for six technology stocks. The FILENAME statement uses the 'STOCKS' FILEREF to identify the name and file directory in which the requested data set will be stored (in this case the directory used by SAS for the WORK library). The %SYSFUNC(GETOPTION(WORK)) statement obtains the physical address of the location of the WORK library. In most cases, this will be the same folder that contains your temporary SAS files.

Program 1.7: Access GitHub Data Repository Using SAS HTTP Request

```
filename stocks "%sysfunc(getoption(WORK))/stocks.sas7bdat";

proc http url="https://github.com/finsasdata/Bookdata/raw/main/stocks.sas7bdat"
        out=stocks
        method ="get";
run;
```

```
proc print data=Stocks(where= (Stock='AMZN' and Date>'31Dec2021'D));
        format volume comma13.;
run;
```

Output 1.7: Daily Stock Prices and Volume of Amazon Inc.



Obs	Date	Stock	Price	Volume
244	31JAN2022	AMZN	$149.57	1,530,000,000
245	28FEB2022	AMZN	$153.56	1,690,000,000
246	31MAR2022	AMZN	$163.00	1,630,000,000
247	29APR2022	AMZN	$124.28	1,470,000,000
248	31MAY2022	AMZN	$120.21	2,260,000,000
249	30JUN2022	AMZN	$106.21	1,770,000,000
250	29JUL2022	AMZN	$134.95	1,340,000,000
251	31AUG2022	AMZN	$126.77	1,170,000,000
252	30SEP2022	AMZN	$113.00	1,210,000,000
253	31OCT2022	AMZN	$102.44	1,460,000,000
254	30NOV2022	AMZN	$96.54	2,040,000,000
255	30DEC2022	AMZN	$84.00	1,550,000,000

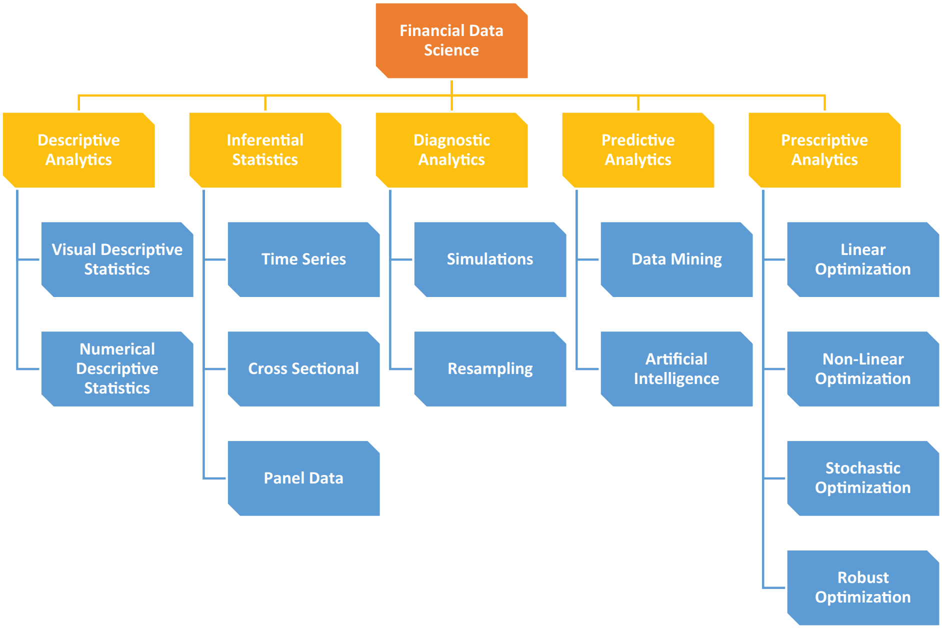
Data Science Concepts and Their Finance Applications

Throughout this book, you will come across various data science concepts and their applications in the finance domain. We will discuss a few of these concepts here. We will also provide basic demonstrations of their applications in SAS. In subsequent chapters, we will conduct in-depth explorations of most of these concepts and provide practical applications of their use in various finance settings.

Descriptive Analytics

Descriptive statistics are essentially the synopses of the main characteristics of the data. Such characteristics include the tally and frequency of the data, and the shape of the distributions as presented using measures of central tendencies and dispersions. In cases where more than one variable is under study, bivariate or multivariate descriptive statistics that present a summary of the relationships between variables might also be presented. Descriptive statistics highlight key attributes of the data and help the researcher formulate the appropriate methodology for executing other statistical and data science techniques. There are two categories of descriptive statistics: numerical descriptive statistics and visual descriptive statistics.

Figure 1.7: Aspects of Financial Data Science



Numerical Descriptive Statistics

Numerical descriptive statistics are usually presented in tabular form and consist of measures of the variable characteristics described above. They might also include results of diagnostic tests that examine various properties of the data, such as skewness, autocorrelations, and stationarity, to name a few. We will discuss many of these statistics in more detail in subsequent chapters of the book.

Financial data are generally presented using a wide variety of numerical descriptive statistics. In the example below, we highlight the SAS program and the results showing simple descriptive statistics of the aggregate financial statement performances of the companies in the S&P 500 index. First, we use the DATA step to compute the annual growth rate in sales (SPSG), earnings per share (EPSG), and dividend payout ratio (DPRG). This is done by calculating the logarithmic return (log difference between current values and lagged values of the same variable, $LOG(\frac{X_t}{X_{t-1}})$, where $X_t = (SPS_t, EPS_t, DPR_t)$. We then calculate the trailing annual price-to-earnings growth (PEG) by dividing the PE ratio by the growth rate of earnings per share $(PEG = PE/EPSG)$. The PEG ratio is a popular measure of how expensive a stock is relative to the growth rate. Higher PEG ratios (above 2) are widely seen as indicative of overvaluation, while lower PEG (below 1) is indicative of undervaluation.

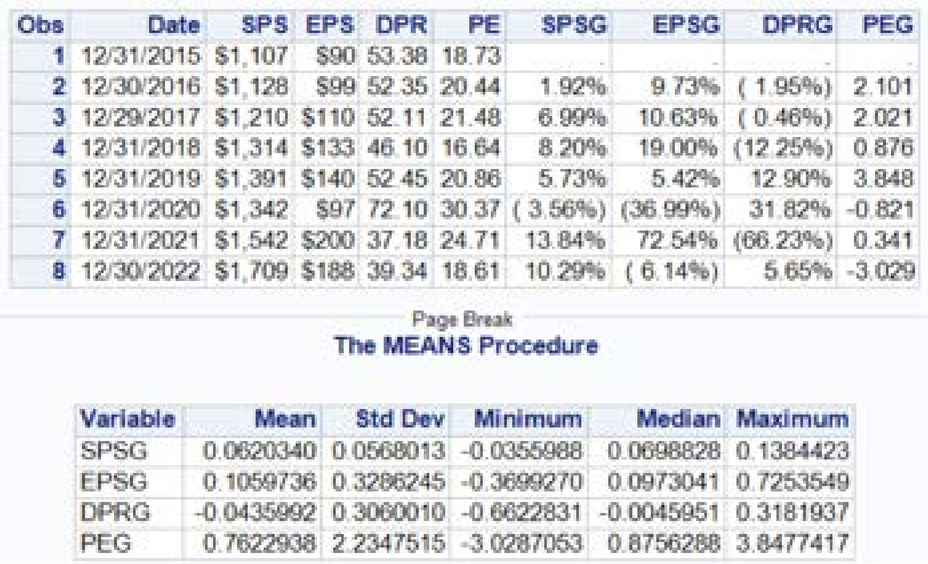
Next, we display the results from our computation using the PROC PRINT statement. We conclude the code by submitting a PROC MEANS statement to compute the mean, standard deviation, minimum, median, and maximum values for each variable.

Program 1.8A: Formatting and Calculating Descriptive Statistics of Aggregate Performance of S&P 500 Companies Using DATA Step and PROC MEANS

```
data NSP500FIN;
      set SP500FIN;
      SPSG=LOG(SPS/LAG(SPS));
      EPSG=LOG(EPS/LAG(EPS));
      DPRG = LOG(DPR/LAG(DPR));
      PEG = PE/(EPSG*100);
label SPSG = 'Sales Growth Rate' EPSG ='Earnings Growth Rate' DPRG='Dividend Payout Ratio
Growth Rate'
             PEG = 'Price-to-Earning Growth Ratio';
format SPSG percent8.2 EPSG percent8.2 DPRG percent8.2 PEG bestd6.;
run;
```

```
proc print data=NSP500FIN;
run;

proc means data=NSP500FIN mean stddev min median max nolabels;
      var SPSG EPSG DPRG PEG;
run;
```

Output 1.8A: Descriptive Statistics of the Aggregate Financial Performance of S&P500 Companies



Obs	Date	SPS	EPS	DPR	PE	SPSG	EPSG	DPRG	PEG
1	12/31/2015	$1,107	$90	53.38	18.73	.	.	.	.
2	12/30/2016	$1,128	$99	52.35	20.44	1.92%	9.73%	(1.95%)	2.101
3	12/29/2017	$1,210	$110	52.11	21.48	6.99%	10.63%	(0.46%)	2.021
4	12/31/2018	$1,314	$133	46.10	16.64	8.20%	19.00%	(12.25%)	0.876
5	12/31/2019	$1,391	$140	52.45	20.86	5.73%	5.42%	12.90%	3.848
6	12/31/2020	$1,342	$97	72.10	30.37	(3.56%)	(36.99%)	31.82%	-0.821
7	12/31/2021	$1,542	$200	37.18	24.71	13.84%	72.54%	(66.23%)	0.341
8	12/30/2022	$1,709	$188	39.34	18.61	10.29%	(6.14%)	5.65%	-3.029

Page Break

The MEANS Procedure

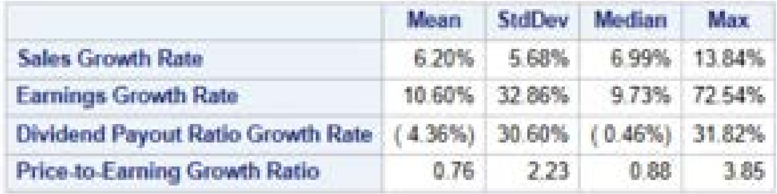
Variable	Mean	Std Dev	Minimum	Median	Maximum
SPSG	0.0620340	0.0568013	-0.0355988	0.0698828	0.1384423
EPSG	0.1059736	0.3286245	-0.3699270	0.0973041	0.7253549
DPRG	-0.0435992	0.3060010	-0.6622831	-0.0045951	0.3181937
PEG	0.7622938	2.2347515	-3.0287053	0.8756288	3.8477417

There are no formatting options for PROC MEANS. To display similar results with your preferred format, use the PROC TABULATE statement.

Program 1.8B: Descriptive Statistics of the Aggregate Financial Performance of S&P 500 Companies

```
proc tabulate data = NSP500FIN;
 var SPSG EPSG DPRG PEG;
 table SPSG*F=percent8.2 EPSG*F=percent8.2 DPRG*F=percent8.2 PEG, mean stddev median max;
run;
```

Output 1.8B: Descriptive Statistics of the Aggregate Financial Performance of S&P 500 Companies



	Mean	StdDev	Median	Max
Sales Growth Rate	6.20%	5.68%	6.99%	13.84%
Earnings Growth Rate	10.60%	32.86%	9.73%	72.54%
Dividend Payout Ratio Growth Rate	(4.36%)	30.60%	(0.46%)	31.82%
Price-to-Earning Growth Ratio	0.76	2.23	0.88	3.85

Visual Descriptive Statistics

Graphical descriptive statistics or data visualization is a data science technique that uses graphical and pictorial depictions to convey stylized facts (such as patterns, trends, and correlations) about the data in a visually compelling and interactive way. Data visualization skills are crucial for financial professionals. Indeed, some financial data are generated at such high frequencies that the only way to quickly communicate and digest the information embedded in them is through visualizations. Hence, financial data is often conveyed through visualizations by the financial media (for example, stock charts, yield curves, and macroeconomic graphs). Visualizations are also used for investment and financial analyses, as well as performance analysis and financial planning.

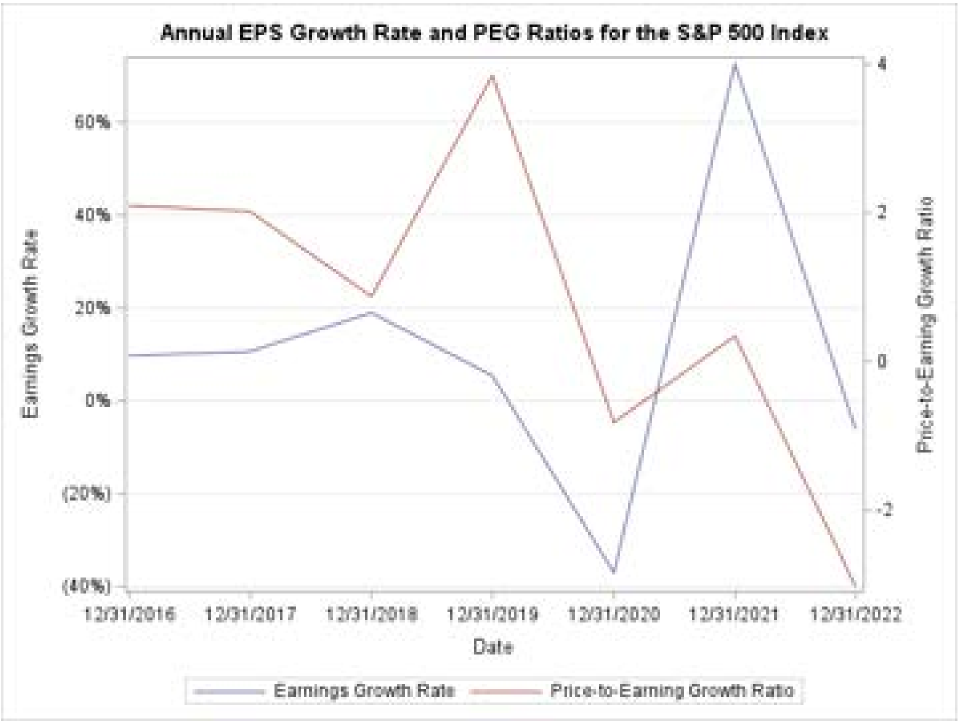
The SAS program shown in Program 1.8C invokes the SGPLOT procedure to graph the time series of the growth rate of aggregate EPS and PEG of the S&P 500 index. In the DATA statement, we use a filter (WHERE) to specify the range of values to plot (post-December 2015). We use the XAXIS statement to specify various configurations for the rendering of the X-axis of the plot.[4]

[4] The SGPLOT procedure is a versatile SAS ODS graphics procedure that can be used to produce a large number of statistical and data visualization charts. We will go into more details about the procedure in Chapter Two. You can learn more about the ODS Graphics procedures by reviewing SAS® 9.4 ODS Graphics: Procedures Guide, Sixth Edition available at https://documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/grstatproc/titlepage.htm.

Program 1.8C: Visualizing the Financial Performance of S&P 500 Companies Using PROC SGPLOT

```
title 'Annual EPS Growth Rate and PEG Ratios for the S&P 500 Index';
proc sgplot data= NSP500FIN (where=(date>'31Dec2015'd));
      series x=Date y=EPSG;
      series x=Date  y=PEG /Y2AXIS;
      xaxis values=('31Dec2016'd to '31Dec2022'd by year);
      yaxis grid;
run;
title;
```

Output 1.8C: Financial Performance of S&P 500 Companies



The graph shown in Output 1.8C suggests that the aggregate valuation of large capitalization stocks appears to move in tandem with the growth in their aggregate earnings. Apparent in the plot are episodes of aggregate overvaluation and undervaluation relative to the growth rate of aggregate earnings.

Inferential Statistics

Most readers of this book would have encountered inferential statistical methods in prior statistics or econometric courses. Remember from the previous section of this chapter that financial econometrics also uses inferential statistical methods. Inferential statistics aim to draw conclusions about events or phenomena that occur in a large group (population) by studying a subset of that group (sample). This is achieved by estimating the value of the unknown characteristics of the population using the sample as the proxy. Hypothesis testing is then performed to validate that the statistics drawn from the sample are reliable estimates of the unknown population characteristics. Hypothesis testing normally would require accepting a set of assumptions and theorems concerning the level of measurement of the variable of interest, the method of sampling, the shape of the population distribution, and the sample size. For example, suppose an equity analyst asserts that over five years, the average growth rates of the return on assets (ROA) of large firms are the same across all sectors of the S&P 500, or a risk manager proclaims that the expected default rate in the bank's auto loan portfolio would not exceed 5% over the life of the loan. These types of assertions are claims that can be tested by inferential statistics.[5]

Let us examine the accuracy of the first claim using inferential statistics. Specifically, we will conduct an ANOVA test (using SAS) of the equality of means to judge whether the five-year average growth rate of the ROAs of firms in the S&P 500 index is the same across the sectors of the index. The Microsoft Excel file SPX_Members.xlsx contains financial data (sales growth, earnings growth, ROA growth, as well as the one-, three-, five-, and ten-year annualized returns) for the 503 firms in the S&P 500 index. Since this data is in Excel format, we will also introduce the SAS code for importing Microsoft Excel files into SAS.

First, let's request the SPX_Members.xlsx file from the GitHub repository by submitting the PROC HTTP statement below. The FILENAME statement uses the 'SPX' FILEREF to identify the name and file directory in which the requested data set will be stored (in this case the directory used by SAS for the WORK library). This will be the SAS temporary folder on your computer or server.

> Modify Program 1.7, which contains the SAS code used for requesting the STOCKS data set from the GitHub repository. SPX_Members.xlsx file might already be in the SAS temporary folder of your computer if you have cloned the entire GitHub repository using Program 1.5.

[5] For those who are new to statistics or need a refresher on basic statistical concepts, SAS offers "Introduction to Statistical Concepts," a free online course in statistics. You can learn more about the course at https://learn.sas.com/course/view.php?id=643.

Program 1.9A: Requesting Excel File from GitHub Data Repository Using SAS HTTP Request

```
filename SPX "%sysfunc(getoption(WORK))/SPX_Members.xlsx";
proc http
      url="https://github.com/finsasdata/Bookdata/raw/main/SPX_Members.xlsx"
      out=SPX
      method ="get";
run;
```

Program 1.9B: Importing Excel File into SAS Using PROC IMPORT

```
/*PROC IMPORT statement below is used to import files from various
systems into SAS. The set of code below imports the SPX_Members.xlsx
file from its current location on the computer into the SPX_Members SAS
datafile in the WORK library*/
proc import out=SPX_Members
      datafile= spx
      dbms= xlsx
      replace;
      getnames= YES;
      sheet= Sheet1;
run;
```

The SPX_Members.xlsx is still in Excel format and would need to be imported into SAS file format using the PROC IMPORT statement below. The DATAFILE statement specifies the name of the Excel data to import into SAS. The OUT statement specifies the SAS name for the imported data set. The DBMS statement informs SAS that the data will be imported from an Excel file format. The REPLACE statement instructs SAS to replace the current version of the SPX data set that might exist in the library with the newly imported version. The GETNAMES statement informs SAS to obtain the variable names from the first row of the Excel file, while the SHEET statement informs SAS to read the data from the specified sheet.

The ANOVA test requires some assumptions about the distribution of the data. These include:

- The five-year growth rates of returns, in general, follow a normal distribution.
- The growth rates of returns have homogenous variance.
- The growth rates of returns were independently sampled.

$$H_0:\ \mu_1 = \mu_2 = \mu_3 \dots \dots . = \mu_6$$

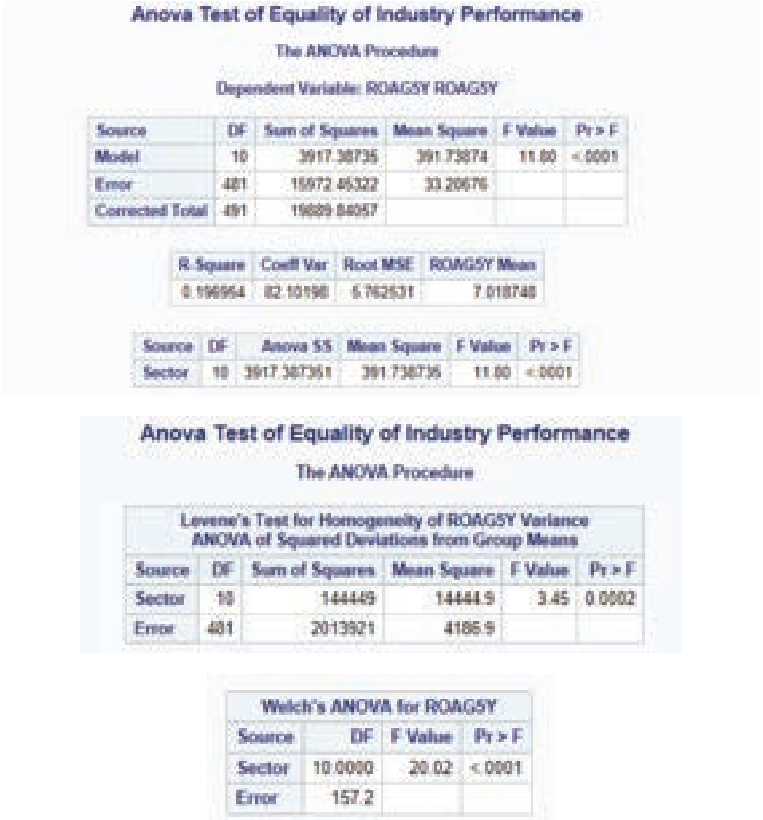
$$H_{01}:\ The\ means\ are\ not\ equal$$

Program 1.9C shows the SAS code for implementing the ANOVA procedure. The ANOVA statement invokes the procedure, followed by the DATA statement that specifies the name of the SAS data set we will be conducting the test on. The CLASS statement specifies the group, which in this case is a group of 11 GIC sectors of the S&P 500 index.

Program 1.9C: Using PROC ANOVA for the Test of Equality of Five-Year Growth Rates Sector ROAs

```
ods graphics on;
proc anova data= SPX_Members;
      title 'Anova Test of Equality of Industry Performance';
      class Sector;
      model ROAG5Y = SECTOR;
      /*Let's also test for equality of variance by including the code
below. If the p-value of the Levene test rejects the null of equal variance,
then the Welch Anova Test of the Equality of means will be used*/
      means SECTOR/hovtest=levene welch;
run;
title;
ods graphics off;
```

Output 1.9C: ANOVA Test for Equality of Five-Year Sector ROA Growth Rates



Anova Test of Equality of Industry Performance

The ANOVA Procedure

Dependent Variable: ROAG5Y ROAG5Y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	10	3917.38735	391.73874	11.80	<.0001
Error	481	15972.45322	33.20676		
Corrected Total	491	19889.84057			

R-Square	Coeff Var	Root MSE	ROAG5Y Mean
0.196964	82.10198	5.762531	7.018748

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Sector	10	3917.387351	391.738735	11.80	<.0001

Anova Test of Equality of Industry Performance

The ANOVA Procedure

Levene's Test for Homogeneity of ROAG5Y Variance
ANOVA of Squared Deviations from Group Means

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Sector	10	144449	14444.9	3.45	0.0002
Error	481	2013921	4186.9		

Welch's ANOVA for ROAG5Y

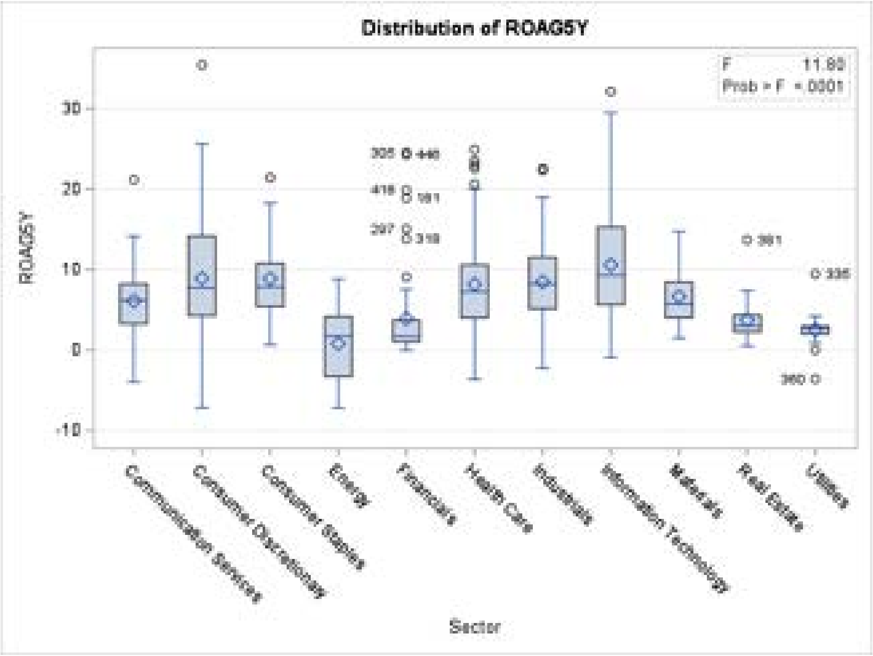
Source	DF	F Value	Pr > F
Sector	10.0000	20.02	<.0001
Error	157.2		

The MODEL statement specifies the numeric dependent variable (ROAG5Y) and the independent effect (SECTORS). The MEANS statement is used to request the computation of the means of the dependent variable for the effect groups. The HOVTEST=LEVENE option is used to request the Levene (1960) test of the homogeneity of variances. While the WELCH option is used to request the Welch (1951) variance-weighted one-way ANOVA.

The p-value of the Levene's test for homogenous variance indicates that we should reject the null. Hence, the results of the Welch ANOVA test (shown below), which adjusts for unequal variances in a one-way ANOVA, will be used. From the Welch ANOVA results, the null of an equal five-year growth rate of ROA for all sectors of the index is rejected. We can then conclude from the results that the five-year average growth rates of ROA and the degrees of their dispersion are statistically not the same across all sectors of the index.

Output 1.9D shows the box plot of the five-year growth rate of ROA for all the sectors. You will notice on the plot that the averages of the growth rates of ROA are not the same across the sectors and that the degrees of dispersion of growth rates are also distinctly different within each sector. The information technology sector appears to have the highest average growth rate, while the consumer discretionary sector appears to have the highest degree of dispersion of the average growth rates. The visualization provides further corroboration for the conclusion that we derived from our inferential test that the equity analyst's assertion is most likely inaccurate.

Output 1.9D: Distribution of Five-Year Sector ROA Growth Rates



Diagnostic Analytics

Diagnostic analytics are used to understand why and how trends, events, and phenomena occur. Although they are often used in conjunction with other types of analytics approaches such as descriptive and predictive analytics, diagnostic analytics are especially useful because they allow us to study how phenomena occur in controlled environments. The insight that we obtained from such studies might establish or rule out the need to conduct further exploration of the problem with other analytics techniques. Financial data scientists use diagnostic analytics to model complex real-world problems, which often involve elements of uncertainty in controlled settings. These models are then used to examine how the system or phenomenon behaves or responds to different inputs or signals.

Simulation is a common methodology used in diagnostic analytics. Simulations are mathematical models that incorporate the essential characteristics of the real-world system or processes to be studied. For example, simulations are used in sensitivity analyses to study why and how outcomes respond to fluctuations in some variables of interest. An index portfolio manager might be interested in how the portfolio will perform over time or when significant events occur in the market. The manager might also be interested in how the portfolio would behave if the assumptions the portfolio manager has made about the distributional properties of the portfolio actually hold in the real world. Banks and other financial services organizations use stress testing (a simulation approach) to also study how their financial structure might be impacted by adverse economic or market conditions.

Simulations will be discussed in more detail in subsequent chapters of the book. For now, let us explore a simple use case of how simulations can be used to examine the assumption about the distributional properties of asset returns. From historical observation, we have determined that the average and standard deviation of the monthly returns on the index from 1985 to 2022 (approximately 457 months) are 0.677% and 4.466%, respectively. If we accept that stock index returns also follow a Gaussian random walk process, we can use simulations to study the theoretical distribution of the monthly returns on the S&P 500 index. A Gaussian random walk process is a stochastic process with a drift component. It is commonly used in simulating the patterns of price evolution for stocks. The discrete version of the equation specification is below.

$$R_t = \mu\Delta t + \sigma\varepsilon\sqrt{\Delta t} \tag{1.1}$$

where $\mu\Delta t$ is the drift component that represents the average monthly rate of return on the index and $\sigma\sqrt{\Delta t}$ is the random or stochastic component in the return process for the index.

For now, let's focus on simulating the stochastic components of the stock return process. In subsequent chapters of the book, we will expand the scope of the simulation and incorporate the drift aspect of the process. If we assume that the stochastic component of the process follows a normal distribution with mean $\mu = 0.677\%$ and standard deviation $\sigma = 4.466\%$, then we can simulate the monthly returns on the S&P 500 using the SAS code in Program 1.10.[6]

[6] It is common in practice to use significantly higher number (in the thousands) of replications to ensure that your results are stable. We will use this approach in subsequent chapters of the book when we cover simulations in more detail. For now, we will use ten replications to keep things simple.

Program 1.10: Simulating Monthly Returns of the S&P 500 Index

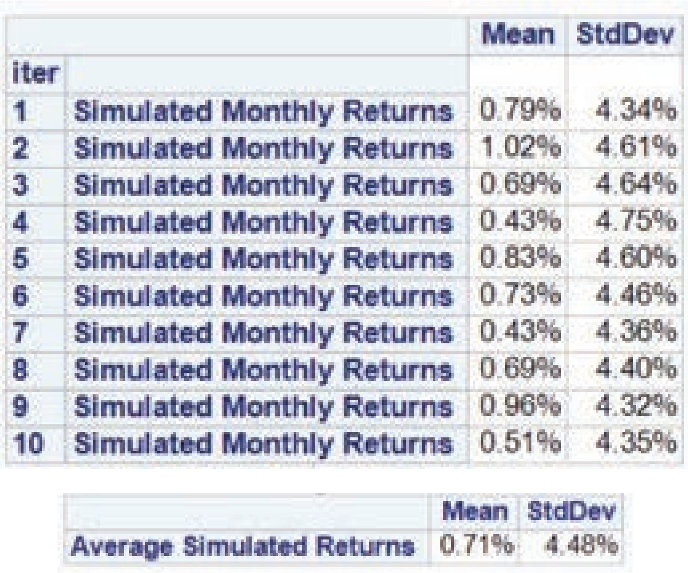
```
%let smean=0.0067658;
%let ssd = 0.04465726;
data SSPX;
   call streaminit(123);
        do iter=1 to 10; /*number of replication*/
                do time = 1 to 457; /*Simulation window*/
                        simret =rand("normal",&smean,&ssd);
                        output;
                end;
        end;
        label Simret='Simulated Monthly Returns';
run;

/* Extracting the Descriptive Statistic for the Simulated Returns*/

proc tabulate data=SSPX;
        class iter;
        var  simret;
        table iter*simret,mean*f=percent8.2 stddev*f=percent8.2;
        table simret='Average Simulated Returns',mean*f=percent8.2 stddev*f=percent8.2;
run;
```

In the code, we stored the mean and standard deviation of the monthly returns in two macrovariables (SMEAN and SSD). The CALL STREAMINIT statement is used to specify the seed values for the subsequent random number generator function, which is invoked using the RAND function. The two DO statements iterate the random number generator over two instances. The instance (1 to 10) is the number of replications to perform, while the second instance (1 to 457) is the number of monthly returns to simulate. Specified in the RAND functions are the two parameters of the normal distributions, which are mean and standard deviation. The OUTPUT statement informs SAS to write the observation to the SAS data set named SSPX. We close each instance of the DO loops with the END statement.

To compute the descriptive statistics, we invoke the TABULATE procedure. We use the ITER variable as a classification variable so that SAS can compute the statistics for each replication of the simulation. The mean and standard deviation statistics for each replication and the entire simulated series are requested using the TABLE statement.

Output 1.10 shows the descriptive statistics produced by invoking the TABULATE statement. Notice the variation in the sample averages computed from each iteration, but a slightly more compact range of values for the standard deviation. You will also notice that the average of the simulated values is very close to the parameters of the distributions that we fed into the RAND function. We will discuss the theory of sampling distribution that supports these findings in Chapter Four.

Output 1.10: Descriptive Statistics from Simulated Monthly Returns of the S&P 500 Index



iter		Mean	StdDev
1	Simulated Monthly Returns	0.79%	4.34%
2	Simulated Monthly Returns	1.02%	4.61%
3	Simulated Monthly Returns	0.69%	4.64%
4	Simulated Monthly Returns	0.43%	4.75%
5	Simulated Monthly Returns	0.83%	4.60%
6	Simulated Monthly Returns	0.73%	4.46%
7	Simulated Monthly Returns	0.43%	4.36%
8	Simulated Monthly Returns	0.69%	4.40%
9	Simulated Monthly Returns	0.96%	4.32%
10	Simulated Monthly Returns	0.51%	4.35%

	Mean	StdDev
Average Simulated Returns	0.71%	4.48%

Predictive Analytics

Predictive analytics is one of the fundamental aspects of data science. It involves the use of data and an assortment of advanced statistical methods to predict future events, behaviors, and trends. More generally, predictive analytics aims to provide decision-makers with the best insights into what will happen in the future. With its origin in data mining, predictive analytics has grown to encompass the use of a wide range of statistical algorithms, data modeling techniques, and artificial intelligence applications such as machine learning and deep learning.

Given the substantial amount of computing power and a vast amount of readily available data to deploy, the applications of predictive analytics in the finance domain are limitless. Predictive analytics is used for algorithmic trading, high-frequency trading, portfolio construction, portfolio risk management, credit risk management, liquidity risk management, fraud prevention, regulatory compliance, and enforcement, to name a few. Indeed, right now, approximately 80 percent of the trading volume on the US stock exchanges is implemented by super-fast computers that have been trained to digest and analyze large amounts of pertinent stock market and economic information for trade signals, which are then implemented at lightning speed to maximize their investment value.[7] Many financial institutions also use predictive analytics to identify suspicious events and prevent fraudulent transactions. Another common

[7] See Amaro (2018).

use of predictive analytics in the financial services industry is for developing credit scoring models for making loan decisions. In subsequent sections of the book, we will showcase some implementations of the financial services applications of predictive analytics models in SAS.

Prescriptive Analytics

Prescriptive analytics involves the use of data and computational algorithms to identify critical factors and the optimal course of action for a particular scenario or business problem. Although its roots are in operations research, prescriptive analytics share similar underpinning with other analytics techniques, in the sense that it is based on unearthing statistical relationships between the choice variables and assessing the impact of the fluctuations in these variables on the decision pattern and possible outcomes under consideration. The aim of prescriptive analytics is to provide decision-makers with the best choice from a variety of possible choices. In some organizations, prescriptive analytics are implemented to extend the insights acquired from prior predictive analytics activities. In finance, prescriptive analytics is mostly used to solve optimization problems. These include linear optimization problems such as revenue and profit maximization, non-linear optimization problems such as portfolio optimization, and stochastic optimization problems, which are common in the risk management and governance domain.

Machine Intelligence and Machine Learning

Although intelligence and learning are used interchangeably, they are not the same. Intelligence is the ability to develop knowledge through cognition, while learning is the acquisition of knowledge through the application of specific methods. For example, by reading and going over the practice exercises in this book, readers will learn about the techniques used in data science. Intelligence encompasses learning as well as other cognitive functions such as perception, attention, memory, and judgment. Therefore, it is evident that learning is a subset of intelligence.

Artificial intelligence and machine learning are data science concepts that are also used interchangeably. However, drawing from the earlier distinction, we can see that they are essentially not the same. Artificial intelligence is the computational mimicry of human intelligence through the development of algorithms that can perceive and adapt to new inputs and synthesize knowledge from them, just like we humans do. Machine learning is a subset of artificial intelligence in which algorithms are trained to sift through large amounts of data and learn from them. Artificial intelligence and machine learning algorithms are used in finance for the predictive analytics cases described in the previous section. Black box trading strategies, which are widely used in finance, rely on algorithms that use machine learning to deduce trading signals from financial and economic data. These signals are often based on complicated and sometimes uninterpretable relationships which the algorithms can decipher from the data by sheer application of brute computing power and complex transformations of the input variables.

Supervised, Unsupervised, and Reinforcement Learning

Machine learning techniques fall into three broad categories: supervised, unsupervised, and reinforcement learning. The main difference between these categories is the process by which the algorithm is trained to solve the prediction or classification problem.

Supervised Learning

In supervised learning, algorithms are trained to discern the predictors of a target outcome or range of outcomes from a set of potential predictors. The algorithm iteratively learns by evaluating obvious and latent relationships between a given set of pre-classified outcomes and its potential predictors. Essentially, the algorithm learns about what variables can predict the target outcome(s) by establishing the existence of relationships between the predictors and the target variable. Although supervised learning is conceptually simple, it is nevertheless a powerful and widely used data science technique in the finance domain. Supervised learning techniques are used for investment decisions, stock prediction models, and credit risk models to name a few. Examples of these algorithms include:

- Decision tree
- Random forest
- Neural network
- Regression models
- Support vector machines

Unsupervised Learning

Unsupervised learning algorithms are fed unlabeled data and tasked with discerning whether commonalities exist between them. What distinguishes this approach from other learning techniques is the minimal level of human intervention that is needed in the iterative process. The algorithm is essentially tasked with sifting through the data to find common patterns or hidden groupings that exist in it without prior knowledge of such structures or patterns. The algorithm learns by iteratively clustering or organizing the variables such that each grouping of the data increasingly would share common attributes or distinguishing features from other groupings.

Financial data scientists use unsupervised learning for dimension reductions and data clustering. Financial and economic variables sometimes share similar underlying properties; therefore, dimension reduction and data clustering are especially useful for big data applications in finance. They help to achieve well-calibrated supervised learning models that are parametrically and non-parametrically efficient. Practical applications of unsupervised learning in finance include portfolio construction, risk governance, and asset pricing. Common types of unsupervised learning include:

- Principal component analysis
- K-means clustering
- Hierarchical clustering

Reinforcement Learning

Reinforcement learning algorithms apply the carrot and stick concept to machine learning. The algorithm learns to discern the relationships in the data through incentive functions, which penalize the algorithm for making wrong predictions and reward it for accurate predictions. The algorithm is typically not told what to predict or classify, but only rewarded or penalized for the accuracy of its predictions or classification. In the same manner as human behavior, the algorithm iteratively learns to solve the prediction or classification problems through the link that the incentive function has to the problem. Learning is fundamentally framed as an optimization problem in which the algorithm seeks to maximize rewards and minimize costs through prediction or classification accuracy. Reinforcement learning is an emerging field in finance, but it has promising future applications in areas such as automated personal financial advisory services (also known as robo-advisors), investment gamification, and trade execution. Common types of reinforcement learning algorithms include:

- Sequential decision-making algorithms
- Value-based decision algorithms
- Policy-based decision algorithms

Parametric Versus Nonparametric Algorithms

Parametric algorithms use a defined set of parameters to model the relationship between the target outcomes and the predictor variables. They typically employ a set of assumptions to define the relationships between the predictors and the target. These assumptions fundamentally then define the mapping functions that the algorithm learns to build the prediction or classification model for the target variable. For example, simple logistic regressions assume a binary target variable, independence between observations, little to no multicollinearity between the predictor variables, and a linear relationship between the predictor variables and the logit of the target variable, to list a few. Nonparametric algorithms do not make specific assumptions about the relationship between the predictor variables and the target variable. Hence, they do not require a specific mapping function and can estimate the unknown function, which could be of any form.

It is important to note that while supervised and unsupervised learning algorithms include those that are parametric and nonparametric, nonparametric algorithms tend to be more frequently seen in the category of unsupervised learning algorithms. An example of a nonparametric type of supervised algorithm is the decision tree algorithm, which we will discuss in a subsequent chapter of the book.

Limitations of Financial Data Science

Data science techniques are powerful tools for solving a myriad of societal and business problems. However, they do have some limitations. We highlight two of these below.

Performance Degradation

As highlighted earlier, the algorithm's ability to solve prediction and classification problems relies on discovering patterns and relationships in the data. In many cases, these patterns and relationships are transitory in nature. Thus, the efficacies of these algorithms in business settings are also often transitory. Performance degradation of predictive algorithms is a widely recognized problem in predictive and machine learning models. This is a particularly challenging problem because there tends to be a lot at stake (at least financially) when these models are applied in financial settings. As the functional efficacy of analytics models degrades, the economic cost of deploying them in financial settings increases. Therefore, the analytic life cycle for financial data scientists is a continuous loop predicated on the need to continuously update or develop new models as existing models lose their potency over time.

Overfitting Versus Underfitting

Figuring out the optimal accuracy of predictive models before deploying them for production continues to be at the forefront of debate in the data science domain. While it is generally agreed that models that are overfitted in the development stage will most likely perform poorly in the production stage, the same notion also applies to under-fitted models. Given the ramifications of such outcomes in financial settings, conducting an honest assessment of model performance is crucial.

Honest model assessment judges the performance of the models using different sets of data than the ones used to train the model. The training data set is used to train the algorithm and as the model is being trained, the validation data set is iteratively used to assess the desired model characteristics. In some cases, a final test data set might also be used to conduct further assessment before the model is deployed for production. We will discuss honest assessment in more detail in Chapter Seven.

You should note that concerns with model fitness in finance transcend the issues of overfitting and underfitting the data. In some finance settings such as stock trading, algorithms are in constant interaction with other algorithms and human traders. However, stock trading is inherently a pseudo-adversarial transaction, and therefore the algorithm might be susceptible to the adversarial tactics of other market actors. (See Nehemya et. al, 2020.) For example, the algorithm might inadvertently fit the model to bad inputs that are intentionally fed to it by other algorithms or market actors. Since trading algorithms learn from the data, they could also inadvertently learn unethical or illegal patterns in market data – these include market manipulation or collusive behaviors. Therefore, the scope of concern with model fit in finance extends beyond overfitting or underfitting but also includes the possibility that the model might be learning the wrong lessons from the data. And with the pervasiveness of black box trading, we might not know that this might be indeed what is happening.

Ethics, Biases, Transparency, and Economic Issues

The widespread applications of data science in virtually every scope of human activity have raised several ethical, social, and economic concerns. As you practice your data science craft, it is important to be cognizant of these issues and how they impact the utility of data science in finance applications. We present a quick summary of some of the ethical, social, and economic issues arising from the widespread use of data science techniques in modern society.

Ethical Issues in Financial Data Science

From an ethical point of view, concerns have been raised about the reliance on algorithms for decision-making. Human decision-making often requires balancing various trade-offs and nuances that algorithms are not naturally equipped to handle. Algorithms are generally designed around the concepts of efficiency and rationality, which sometimes conflict with observed human behaviors, emotions, and preferences such as altruism and fairness, to name a few.

One area of ethical concern in the use of data science in finance is the impact of algorithms on the structure and functioning of financial markets. Critics of algorithmic trading argue that the high levels of volatility that are now a common feature of the financial markets are due to the pervasiveness of algorithm-based trading strategies.[8] Others have argued that algorithmic trading provides little utility in terms of price discovery and allocative efficiency while extracting value from investors who do not have access to similar tools (Yadav, 2015). In May 2010, a self-taught stock trader operating out of his bedroom triggered a $1 trillion momentary crash in the US stock market. In May 2022, a trader at Citigroup's London office mistakenly added an extra zero to the trade order and in a split second, caused the entire stock market across Europe to crash by as much as 8% (over $300 billion) in a matter of minutes.[9] In reality, the amplifying forces that cause

[8] Articles in the financial news media periodically feature arguments along this line. See for example, "Volatility: How 'Algos' Changed the Rhythm of the Market" in the January 9th, 2019, edition of Financial Times and "A Down Day on the Markets? Analysts Say Blame the Machines" in the February 8th, 2018, edition of The Washington Post.

[9] On August 24th, 2015, another algorithm-related flash crash occurred in the first 15 minutes of trading. This crash is speculated to have been caused by overnight drying-up of liquidity. Algorithms responded to the dislocation in the market by halting trades, thereby further exacerbating the liquidity problem. More on the cause of this crash can be found at https://www.cnbc.com/2015/09/25/what-happened-during-the-aug-24-flash-crash.html.

the actions of these single individuals to result in such devastating financial outcomes are the innumerable number of trading algorithms that have been deployed in financial markets across the globe.

Bias

Algorithms learn from the information provided to them, and if the information is biased, the predictions or classifications made by the algorithm might also be biased as well. Along this line, some have argued that algorithms actually exacerbate social problems because they learn and then reinforce existing biases from the societal data. For example, a 2018 study by professors from the University of California Berkeley found that while algorithmic-based mortgage decisions result in less discriminatory (in terms of loan approval) outcomes for minority borrowers than face-to-face underwriting, they still result in significantly higher loan rates for minority borrower compared to white borrowers, after controlling for all other risk factors (such as income, credit scores, and loan amount) that go into loan underwriting process (Bartlett et al., 2019).

Transparency

One of the costs associated with the increasing sophistication of algorithms that are being developed is the loss of interpretability. Algorithms are essentially becoming black boxes that accept inputs and spew out output, which the end user must accept with blind faith. Interpretable models are vital in decision-making because they allow the decision-maker to link the decision outcomes to observable, repeatable, and assessable functions of the inputs. It allows the user to know why they are following the set of actions that the algorithm is recommending. Machines are not perfect, they sometimes malfunction, and without transparency, we might not know that such malfunction has occurred until it is too late. Concerning our earlier discussion of the flash crashes, it is useful to think of the actions of these traders as inputs, which eventually led all these trading algorithms to react in ways that moved the markets precipitously. However, given that we, in general, still lack the full understanding of how the black boxes of algorithmic trading work, the intervention by regulators so far has been limited to implementing circuit breakers to prevent the problems from getting worse when they do occur.

Economics

Many of the structural changes that we have seen in the global economy over the past three decades can be traced to advancements in computing. If we are to conceptualize what the future economy would look like, given our recent history, the conclusion will be that a computational economy driven by advanced data sciences is what the future portends. However, this future raises a lot of questions that data scientists and policymakers would need to grapple with. For example, What would happen to people whose jobs or industries are disrupted or replaced by algorithms? How would policymakers respond to economic inequality, which some argue has been made worse by algorithms? (Zatko, 2022) How much control and autonomy are we as a society willing to surrender to algorithms? How would the regulatory and legal landscape evolve to potentially new and disruptive technologies? These are a few of the economic dilemmas that society will need to address as we become more reliant on data science to drive our economies.

Regulatory Landscape

Financial institutions are highly regulated entities and over the years, these organizations have become more reliant on data science to drive their business process and regulatory compliance reporting, thereby creating a nexus of regulatory concerns for how data science is used in the industry. In this book, we will explore two areas where data science is used by financial institutions for regulatory reporting purposes. Specifically, we will focus on how data science is used for market and credit risk modeling.

Risk modeling is in itself a risky endeavor because it entails making forecasts about future risk outcomes. For financial institutions, such forecasts are often associated with regulatory mandates concerning capital provisioning, risk governance and budgeting, and strategic business choices. Therefore, extra care is warranted when employing data science techniques in financial services organizations. Indeed, all financial services organizations with regulatory reporting mandates are required to adhere to a framework for model risk management. This mandate was set to ensure that their modeling processes are in line with best practices and that they present accurate pictures of the financial standing of the organization for regulatory purposes.

For market risk, we will explore examples of value-at-risk (VaR) implementations in SAS. For credit risk, we will explore the implementation of various reportable measures relating to credit portfolios in SAS. These include modeling the credit exposures, default probabilities, and their determinants for portfolios of credit obligations.

Exercises

1. The text file Portfolio01.txt contains the stock tickers, number of shares, cost basis, ending beats, and the betas of nine stocks in a concentrated portfolio.

```
Ticker Shares Price TotRet Beta
AAPL 5907.17 11.85 405 1
BA 4002.46 32.48 72.92 1.14
BAC 4835.16 22.75 5.55 2.07
CAT 7115.75 21.08 90.2 2.01
GE 1707.07 29.29 17.75 1.67
IBM 820.94 109.63 183.17 0.61
LMT 2910.82 37.79 80 0.7
MSFT 5245.41 26.69 25.79 1.03
PNC 3465 43.29 57.34 1.36
```

a. Write a SAS program to create a temporary SAS data set (**Portfolio01**) using the data in Portfolio01.txt.

b. Using the created **Portfolio01** data set, create a new variable InitValue that calculates the initial dollar investment in each stock (multiply Shares by Cost) and a variable that calculates the final value (FinValue) of the dollar investment in each stock (multiply Shares by Price).

c. Create a new variable (TotRet) that calculates the ten-year return on each stock.

d. The total initial dollar amount invested in the portfolio is one million dollars. Use this information to create a new variable (InitWeight) that calculates the initial portfolio weights in each stock (InitValue/$1,000,0000).

e. Assign the following formats and labels to each variable.

Variable	Format	Label
Shares	Comma10.2	Number of Shares
Cost	Dollar10.2	Cost Basis
Price	Dollar10.2	Current Price
Beta	Bestd6.2	Beta
TotRet	Percent8.2	Holding Period Return
InitValue	Dollar13.2	Initial Investment Value
FinValue	Dollar13.2	Final Investment Value
InitWeight	Percent8.2	Initial Weight

f. Use the PROC PRINT statement to display your completed Portfolio01 data set using the format and variable labels.

   i. Which stocks had the highest and lowest initial values?

   ii. Which stocks had the highest and lowest final values?

   iii. Over the ten years, did cheaper stocks perform better than more expensive stocks?

   iv. Over the ten years, did stocks with higher initial shares (weights) perform better than those with a lower initial number of shares (weights)?

   v. Did stocks with higher betas perform better than stocks with lower betas?

2. Let's conduct a portfolio-level analysis of our Portfolio01 data set.

a. Write a SAS program to calculate the average, sum, minimum, maximum, and median number of shares, costs, prices, and holding period returns for the stocks in the portfolio. If your result is unformatted, you probably used PROC MEANS. Create the same result using PROC TABULATE and specify the correct format for each statistic as shown in the table above.

Hint: Include the following modification to the TABLE statement in PROC TABULATE.

```
Shares*F=comma10.2 Cost*F=dollar10.2 Price*F=dollar10.2
Beta*F=bestd6.2 initvalue*F=dollar13.2 FinValue*F=dollar13.2
TotRet*F=percent8.2,mean sum median min max;
```

   i. What is the average number of shares, costs, prices, beta, initial values, and final values of the stocks in the portfolio?
   ii. What was the ending value of the portfolio? Given the initial value of the portfolio, what were the dollar and percent holding period returns on the portfolio?
   iii. Which stocks contributed the most value to the portfolio? Which contributed the least?

b. Now write a SAS program to calculate the holding period return and the beta of the portfolio. Compare the result to the calculations you performed in Part A.

> **Hint:** Include the following modification to the TABLE statement for PROC TABULATE.
>
> ```
> Shares*F=comma10.2 Cost*F=dollar10.2 Price*F=dollar10.2
> initvalue*F=dollar13.2 FinValue*F=dollar13.2 TotRet*F=percent8.2 ,mean;
> ```

3. A macro analyst at a boutique investment fund is evaluating the impact of various economic factors on the stock market performance. The analyst collects data on the monthly percent change in the following variables from the Federal Reserve Bank of St. Louis's FRED economic database[10] and compiles them into the Freddata01 SAS file. (The data set is available in the book's GitHub Repository: https://github.com/finsasdata/Bookdata/raw/main/freddata01.sas7bdat). The variable names and their labels are shown in the table below.

UNRATE	Unemployment Rate, Seasonally Adjusted
PCE	Personal Consumption Expenditures, Seasonally Adjusted Annual Rate
AAA10Y	Moody's Seasoned Aaa Corporate Bond Yield Relative to Yield on 10-Year Treasury Constant Maturity
HOUST	New Privately-Owned Housing Units Started: Total Units, Seasonally Adjusted Annual Rate
ICSA	Initial Claims, Seasonally Adjusted
T10Y3M	10-Year Treasury Constant Maturity Minus 3-Month Treasury Constant Maturity, Not Seasonally Adjusted
PPIACO	Producer Price Index by Commodity: All Commodities, (Index 1982=100), Not Seasonally Adjusted
WILLLRGCAP	Wilshire US Large-Cap Total Market Index, Not Seasonally Adjusted

[10] The Federal Reserve Bank of St. Louis' FRED economic database is one of the largest free databases of US and international economic and financial time series. Visit https://fred.stlouisfed.org/ to learn more about it and to request your own data for further exploration.

WILLMIDCAP	Wilshire US Mid-Cap Total Market Index, Not Seasonally Adjusted
WILLSMLCAP	Wilshire US Small-Cap Total Market Index, Not Seasonally Adjusted
WILL5000IND	Wilshire 5000 Total Market Index, Not Seasonally Adjusted
UMCSENT	University of Michigan: Consumer Sentiment, Percent Change

a. Write a SAS procedure (PROC CORR) to examine the relationships between these macroeconomic variables and the four market indices.

> **Hint:** This can also be executed using the TASK menu in both Enterprise Guide and SAS Studio.

i. Which economic variables have statistically significant positive and negative correlations with the four market indices? Why do some variables have positive and others negative relationships with the market indices?

ii. Are there any differences in the correlations between the economic variables and the four market indices? What could explain these differences?

iii. Which economic variables have statistically significant positive and negative correlations with other economic variables? Why are these economic variables correlated?

b. Write a SAS Program (PROC SGPLOT) to graph the relationship between the Wilshire 5000 Total Market Index (WILL5000IND) and consumer sentiments (UMSCENT).

> **Hint:** This can also be executed using the TASK menu in both Enterprise Guide and SAS Studio.

i. Repeat the same graph for the Wilshire Large Cap, Mid Cap, and Small Cap indices.

4. A portfolio analyst at the same investment fund wants to examine if stock index returns truly follow a normal distribution by analyzing the distributional properties of the four Wilshire market indices (WILLLRGCAP, WILLMIDCAP, WILLSMLCAP, and WILL5000IND). Using the Freddata01 SAS data file, write a SAS program that uses PROC UNIVARIATE to analyze each index return to assess whether the distribution is truly normal. Include both tables and graphs in your results.

> **Hint:** This can also be executed using the TASK menu in both Enterprise Guide and SAS Studio. Look for the DISTRIBUTION ANALYSIS Menu.

a. What are the mean, median, and standard deviation of the monthly returns for the indices?
b. Using the Moments, Quantiles, and Extreme Observations Table as references, characterize the shape of the distribution of the monthly returns (is it symmetric or asymmetric, compact or dispersed).
c. Modify your PROC UNIVARIATE statement to include a histogram. Also, superimpose the Normal (expected) and Kernel (fitted) densities of the returns on the histogram. Do these graphs support the notion that monthly returns on the Wilshire indices are normally distributed?
d. What are the investment implications (in terms of portfolio behaviors and performance measurement) of the statistical features observed in parts a, b, and c for investors who hold the Wilshire indices in their portfolios?

5. **Case Analysis**

Sally Smith graduated a year ago with a Ph.D. in financial engineering from a prestigious university. Upon graduation, she immediately landed a lucrative job as a quantitative data analyst at one of New York City's Wall Street banks. Her main responsibility is to develop machine learning models that can sift through large volumes of social media posts and macroeconomic news to draw trading signals for the bank's equity trading desk. Her annual compensation is partly tied to how much profit the trading desk can earn using her models. She spent the first six months developing various machine-learning models that can decipher which social media posts convey positive and negative sentiments about a few stocks in the technology industry. While developing the models, she noticed that specific sets of social media posts appear to consistently convey sentiments that are in line with the future movement in the price of some stocks. However, she is unable to tell if her models are incorporating the social media posts in the trading signals they yield for the equity trading desk because they are based on black box algorithms. She's worried that the persons behind the social media handles could be using insider information and trying to manipulate the market with their posts. Sally also recently discovered another algorithm that is in direct conflict with her algorithms (taking opposite trades every time Sally's algorithm trades). The adversarial algorithm does not appear to have much impact on the predictive performance of Sally's algorithm.

Discussion Questions

a. Is it ethical for the bank to tie Sally's compensation to the profitability of her algorithm?
b. What ethical responsibilities does Sally have as a data scientist regarding the social media posts?
c. What legal responsibilities does she have regarding the same issue?
d. How should Sally proceed with the development of her machine-learning models without compromising her values?
e. What should Sally do to address the issue of the adversarial algorithm?