



الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming

Module 1: Orientation



Contents

!

1. Course Description.
2. Course Overview.
3. Course's main objectives.
4. Textbooks.
5. Course Learning Outcomes.



Course Description



Course Description |

!

This course is intended to be an introduction to data science concepts and techniques. The content is divided into two parts. The first part is dedicated to discovering fundamentals on the standards and practices for collecting, organizing, managing, exploring, and using data.

The second part is dedicated to go through the needed techniques to gain insights of the data using Python.

Course Overview.



Course Overview

!

!

Many important decisions made by individuals and society are or should be data driven. Therefore, understanding the fundamentals of data science is essential for right actions.

The primary goal of this course is to have the experience with manipulating, analyzing, and presenting data which are increasingly important.

The second goal is to be able to analyze a problem with an aim to conduct analyses with Python.

Textbooks.



Textbooks

!

Required book 1:

Lillian Pierson, Data Science, 3rd Edition, 2021, ISBN: 978-1-119-81155-8 (35 USD)

Required book 2:

John C.Shovic and Alan Simpson, Python All-in-One, 2nd edition 2021, ISBN: 978-1-119-78760-0 (40 USD)



Course Learning Outcomes



Course Learning Outcomes

!

!

Identify and describe the methods and techniques commonly used in data science

Apply data preprocessing tools

Develop Python code elements

Analyze and evaluate results of data science solutions

Demonstrate how data analysis, modeling, machine learning, and statistical computing can be utilized in an integrated competence



Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming
Module 2:
Wrapping Your Head around Data Science



Contents

1. Seeing Who Can Make Use of Data Science
2. Inspecting the Pieces of the Data Science Puzzle
3. Exploring Career Alternatives That Involve Data Science



Weekly Learning Outcomes

1. Deploying data science methods across various industries
2. Piecing together the core data science components
3. Identifying viable data science solutions to business challenges
4. Exploring data science career alternatives



Required Reading

1. Chapter 1. Wrapping Your Head around Data Science
(Lillian Pierson, Data Science, 3rd Edition, 2021)

Videos

- ¥ Data Science In 5 Minutes | Data Science For Beginners | What Is Data Science?
<https://youtu.be/X3paOmcrTjQ>
- ¥ Data Analyst vs Data Scientist vs Data Engineer Picking The Role That'll Make You
HAPPY & PROSPEROUS
<https://youtu.be/DVOvRM7r3gM>



Introduction



Introduction |

- ¥Data is coming from every computer, every mobile device, every camera, and every imaginable sensor.
- ¥Data is generated in every social media interaction we humans make, every file we save, every picture we take, and every query we submit.
- ¥Data is even generated when we do something as simple as a favorite search engine for directions to the closest ice cream shop.
- ¥Data is streaming from almost every activity that takes place in both the digital and physical worlds.

Introduction |

- ¥Data is structured, semistructured, and unstructured.
- ¥The question is : What's the point of all this data? Why use valuable resources to generate and collect it?
- ¥Specialists known as data engineers are constantly finding innovative and powerful new ways to capture, collate, and condense unimaginably massive volumes of data.
- ¥And other specialists, known as data scientists, are leading change by deriving valuable and actionable insights from that data.

Introduction |

- ¥ In its truest form Data Science represents the optimization of processes and resources. Data science produces data insights.
- ¥ Using data science insights is like being able to see in the dark. For any goal you can imagine, you can find data science methods to help you predict the most direct route from where you are to where you want to be.

Welcome to the magic of Data Science



1. Seeing Who Can Make Use of Data Science



Seeing Who Can Make Use of Data Science

- ✖ The terms data science and data engineering are often misused and confused.
- ✖ Data science is the computational science of extracting meaningful insights from raw data and then effectively communicating those insights to generate value.
- ✖ Data engineering is an engineering domain that's dedicated to building and maintaining systems that overcome data processing bottlenecks and data handling problems for applications that consume, process, and store large volumes, varieties, and velocities of data.

Seeing Who Can Make Use of Data Science

Three data varieties:

- ¥Structured Data that is stored, processed, and manipulated in a traditional relational database management system (RDBMS). Example: a MySQL database
- ¥Unstructured Data that is commonly generated from human activities and doesn't fit into a structured database format. Example: email documents
- ¥Semistructured Data that doesn't fit into a structured database system but is nonetheless organizable by tags that are useful for creating a form of order and hierarchy in the data. Example: XML and JSON files.

Seeing Who Can Make Use of Data Science

- ¥ It used to be that only large tech companies implement data science methodologies to optimize and improve their business, but that's not been the case for quite a while now. The proliferation of data has created a demand for insights.
- ¥ Data and the need for data-informed insights are found everywhere.
- ¥ Because organizations of all sizes are beginning to recognize that they're immersed in a sink-or-swim, data-driven, competitive environment, data know-how has emerged as a core and requisite function in almost every line of business.

Seeing Who Can Make Use of Data Science

- ¥The fact is, in order to stay relevant, you need to take the time and effort to acquire the skills that keep you current.
- ¥Who can use data science? You can. Your organization can. Your employer can. Anyone who has a bit of understanding and training can begin using data insights to improve their lives, their careers, and the well-being of their businesses.
- ¥With data insights, however, people now have access to the predictive vision that they need to truly drive change and achieve the results they want.

1. Inspecting the Pieces of the Data Science Puzzle



Inspecting the Pieces of the Data Science Puzzle

- ¥To practice data science, in the true meaning of the term, you need the analytical know-how of math and statistics, the coding skills necessary to work with data, and an area of subject matter expertise.
- ¥Nowadays, it's almost impossible to differentiate between a proper data scientist and a subject matter expert (SME) whose success depends heavily on their ability to use data science to generate insights.

Inspecting the Pieces of the Data Science Puzzle

¥The key components that are part of any data science role:

- | Collecting, querying, and consuming data
- | Applying mathematical modeling to data science tasks
- | Deriving insights from statistical methods
- | Coding, coding, coding Ñ itÕs just part of the game
- | Applying data science to a subject area
- | Communicating data insights

Inspecting the Pieces of the Data Science Puzzle

| Collecting, querying, and consuming data

- ✖ Data engineers have the job of capturing and collating large volumes of structured, unstructured, and semi structured data
- ✖ Again, data engineering tasks are separate from the work that performed in data science, which focuses more on analysis, prediction, and visualization. Despite this distinction, whenever data scientists collect, query, and consume data during the analysis process, they perform work similar to that of the data engineer.

Inspecting the Pieces of the Data Science Puzzle

| Collecting, querying, and consuming data

- ¥A data scientist can work from several datasets that are stored in a single database, or even in several different data storage environments.
- ¥No matter how the data is combined or where it's stored, if you're a data scientist, you almost always have to query data.
- ¥Whether you're using a third party application or doing custom analyses by using a programming language such as R or Python, you can choose from the following universally accepted file formats:

Inspecting the Pieces of the Data Science Puzzle

Collecting, querying, and consuming data

- ! Comma-separated values (CSV): Almost every brand of desktop and web-based analysis application accepts this file type.
- ! Script: These script files end with the extension `.py` (Python) or `.r` (R).
- ! Application: Excel is useful for quick and easy analyses on small to medium size datasets.
- ! Web programming: If you're building custom, web-based data visualizations, you may be working in D3.js or data-driven documents, a JavaScript library for data visualization.

Inspecting the Pieces of the Data Science Puzzle

| Applying mathematical modeling to data science tasks

- ¥Data science relies heavily on a practitioner's math skills: Precisely because these are the skills needed to understand your data and its significance.
- ¥These skills can be used to carry out predictive forecasting, decision modeling, and hypotheses testing.
- ¥Mathematics uses deterministic methods to form a quantitative(or numerical) description of the world.
- ¥Data scientists use mathematical methods to build decision models, generate approximations, and make

Inspecting the Pieces of the Data Science Puzzle

| Deriving insights from statistical methods

✚ In data science, statistical methods are useful for better understanding your data's significance, for validating hypotheses, for simulating scenarios, and for making predictive forecasts of future events.

✚ If you want to go places in data science, though, take some time to get up to speed in a few basic statistical methods, like linear and logistic regression, naïve Bayes classification, and time series analysis.

Inspecting the Pieces of the Data Science Puzzle

■ Coding, coding, coding - it's just part of the game

- ¥ Coding is unavoidable when you're working in data science
- ¥ Your code will instruct the computer in how to manipulate, analyze, and visualize your data.
- ¥ Programming languages such as Python and R are important for writing scripts for data manipulation, analysis, and visualization.
- ¥ SQL, on the other hand, is useful for data querying.
- ¥ Finally, the JavaScript library D3.js is often required for making cool, custom, and interactive web-based data visualizations.

Inspecting the Pieces of the Data Science Puzzle

| Applying data science to a subject area

- ¥ Many statisticians have cried out, "Data science is nothing new - it's just another name for what we've been doing all along!"
- ¥ In fact, data scientists often use computer languages not used in traditional statistics and take approaches derived from the field of mathematics.
- ¥ The main point of distinction between statistics and data science is the need for subject matter expertise

Inspecting the Pieces of the Data Science Puzzle

| Applying data science to a subject area

- ¥Data scientists, should have a strong subject matter expertise in area in which they're working.
- ¥Data scientists generate deep insights and then use their domain specific expertise to understand exactly what those insights mean with respect to the area in which they're working.
- ¥An example of coupling data science skills with area of expertise: Clinical informatics scientist, that combines healthcare expertise with data science skills to produce personalized healthcare treatment plans (more examples in the book)

Inspecting the Pieces of the Data Science Puzzle

| Communicating data insights

- ¥A data scientist must have excellent verbal communication skills.
- ¥If a data scientist can't communicate, all the knowledge and insight in the world does nothing for the organization.
- ¥Data scientists need to be able to explain data insights in a way that staff members can understand.
- ¥Data scientists need to be able to produce clear and meaningful data visualizations and written narratives.
- ¥Data scientists must be creative and pragmatic in their means & methods of communication.



1. Exploring Career Alternatives That Involve Data Science



Exploring Career Alternatives That Involve Data Science

The data implementer

- ¥The main task is to build data and artificial intelligence (AI) solutions.
- ¥Attention should be given to detail that naturally helps you in coding up innovative solutions that deliver reliable and accurate results.
- ¥A project starts with a simple request and some messy data, but through perseverance and brainpower, they're turned into clear and accurate predictive data insights.
- ¥If you're a data implementer, math and coding are your power.

Exploring Career Alternatives That Involve Data Science

The data leader

- ¥Data leader leads teams and project stakeholders through the process of building successful data solutions.
- ¥ The difference between data implementers and data leaders is leaders generally love data science for the incredible outcomes that it makes possible.
- ¥They have a deep passion for using their data science expertise and leadership skills to create tangible results.
- ¥Data leaders love to collaborate with smart people across the company to get the job done right.

Exploring Career Alternatives That Involve Data Science

I The data entrepreneur

- ¥ If you're a data entrepreneur, your secret superpower is building up businesses by delivering exceptional data science services or products.
- ¥ A data entrepreneur has many overlapping traits and a greater affinity for either the data implementer or the data leader, but with one important difference: Data entrepreneurs crave the creative freedom that comes with being a founder.
- ¥ The goal is to create a vision for a business and then use data science expertise to guide the business to turn that vision into reality.



Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming
Module 3:
Tapping into Critical Aspects of Data Engineering



Contents

1. Defining Big Data and the Three Vs
2. Identifying Important Data Sources
3. Grasping the Differences among Data Approaches
4. Storing and Processing Data for Data Science



Weekly Learning Outcomes

1. Unraveling the big data story
2. Looking at important data sources
3. Differentiating data science from data engineering
4. Storing data on-premise or in a cloud
5. Exploring other data engineering solutions



Required Reading

1. Chapter 2. Tapping into Critical Aspects of Data Engineering (Lillian Pierson, Data Science, 3rd Edition, 2021)

Videos

Sources of Data

<https://youtu.be/bbgDaRtKRQE>

- ¥ What Is Cloud Computing And How It Is Enabling The Big Data Economy

<https://youtu.be/SdvF9ohvmpw>



Introduction



Introduction |

- ¥Most normal people aren't aware of what data really is or how it's used to improve people's lives.
- ¥This chapter tells the full story about big data, explains where big data comes from and how it's used.
- ¥We will outline the roles that machine learning engineers, data engineers, and data scientists play in the modern data ecosystem and how data science to improve business performance.
- ¥Fundamental concepts related to storing and processing data for data science will be introduced.

1. Defining Big Data and the Three Vs



Defining Big Data and the Three Vs

- ¥ Big Data Ñ a term that characterizes data that exceeds the processing capacity of conventional database systems because it's too big, it moves too fast, or it lacks the structural requirements of traditional database architectures.
- ¥ In order to use big data, you need to have data storage and processing capabilities that are available only if you invest in a Hadoop cluster.

Defining Big|Data and the Three Vs

- ✖ Hadoop is a data processing platform that is designed to reduce big data into smaller datasets that are more manageable for data scientists to analyze.
- ✖ Hadoop is, and was, powerful at satisfying one requirement: batch-processing and storing large volumes of data.

Defining Big|Data and the Three Vs

- ¥ If companies want to stay competitive, they must be proficient and expert at infusing data insights into their processes, products, as well as their growth and management strategies.
- ¥ This is especially true in light of the digital adoption explosion that occurred as a direct result of the COVID-19 pandemic
- ¥ Whether your data volumes rank on the terabyte or petabyte scales, data-engineered solutions must be designed to meet requirements for the data's intended destination and use.

Defining Big Data and the Three Vs

- Three characteristics – also called the three Vs – define big data: Volume, Velocity, and Variety.
- Because the three Vs of big data are continually expanding, newer, more innovative data technologies must continuously be developed to manage big data problems.
- In a situation where you're required to adopt a big data solution to overcome a problem caused by your data's velocity, volume, or variety, you have moved out the domain of regular data – you have a big data problem on your hands

Defining Big|Data and the Three Vs

| Dealing with data Volume

- ¥The lower limit of big data volume starts as low as 1 terabyte and it has no upper limit. If your organization owns at least 1 terabyte of data, that data technically qualifies as big data.
- ¥Big data is composed of huge numbers of very small transactions that come in a variety of formats.
- ¥Data engineers have the job of aggregating it, and data scientists have the job of analyzing it.

Defining Big|Data and the Three Vs

| Handling data Velocity

- ¥A lot of big data is created by using automated processes.
- ¥In engineering terms, data velocity is data volume per unit time. Big data enters an average system at velocities ranging between 30 kilobytes (K) per second to as much as 30 gigabytes (GB) per second.
- ¥High-velocity, real-time moving data presents an obstacle to timely decision-making. The capabilities of data handling and data processing technologies often limit data velocities.

Defining Big|Data and the Three Vs

| Dealing with data Variety

- ¥Big data gets even more complicated when you add unstructured and semistructured data to structured data sources.
- ¥Heterogeneous, high-variety data is often composed of any combination of graph data, JSON files, XML files, social media data, structured tabular data, weblog data, and data generated from user clicks on a web page, known as click-streams.

Defining Big Data and the Three Vs

| Dealing with data Variety

- ✖ Structured data can be stored, processed, and manipulated in a traditional relational database management system (RDBMS).
- ✖ Unstructured data comes completely unstructured: it's commonly generated from human activities and doesn't fit into a structured database format.
- ✖ Semistructured data doesn't fit into a structured database system, but is nonetheless structured, by tags that are useful for creating a form of order and hierarchy in the data.



1. Identifying Important Data Sources



Identifying Important Data Sources

- ¥ Vast volumes of data are continually generated by humans, machines, and sensors everywhere.
- ¥ Typical sources include data from social media, financial transactions, health records, click-streams, log files, and the Internet of things

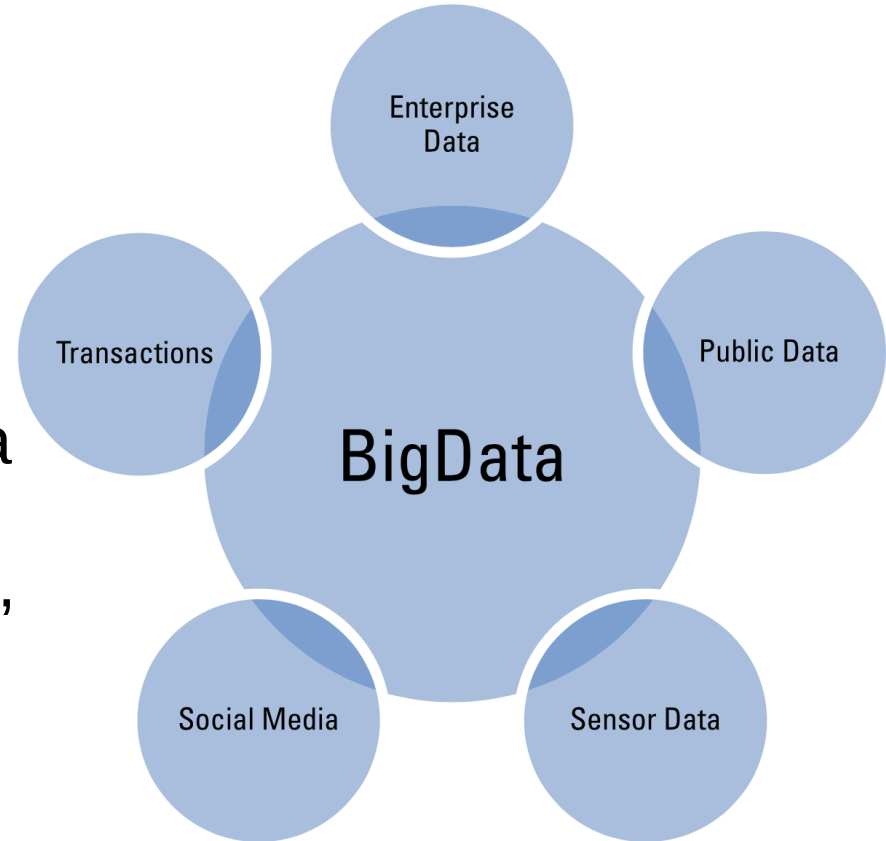


FIGURE 2-1:

Popular sources of big data.



1. Grasping the Differences among Data Approaches



Grasping the Differences among Data Approaches

- ¥Data science, machine learning engineering, and data engineering cover different functions within the big data paradigm.
- ¥Huge velocities, varieties, and volumes of structured, unstructured, and semistructured data are being captured, processed, stored, and analyzed using a set of techniques and technologies.
- ¥Although the terms data science and data engineering are often used interchangeably, they are distinct domains of expertise. Following we will introduce concepts that are fundamental to data science and data engineering, as well as the hybrid machine learning engineering role.

Grasping the Differences among Data Approaches

| Defining Data Science

- ¥Data science is the scientific domain dedicated to knowledge discovery via data analysis.
- ¥Data science is a vast and multidisciplinary field. To call yourself a true data scientist, you need to have expertise in math and statistics, computer programming, and your own domain-specific subject matter.
- ¥Data science methods can provide more robust decision-making capabilities in business and in science.

Grasping the Differences among Data Approaches

| Defining Data Science

¥Using data science skills, you can do cool things like the following:

- ! Use machine learning to optimize energy usage
- ! Optimize tactical strategies to achieve goals in business and science.
- ! Predict for unknown contaminant levels from sparse environmental datasets.
- ! Design automated theft and fraud prevention systems to detect anomalies.

Grasping the Differences among Data Approaches

| Defining machine learning engineering

- ¥Machine learning is the practice of applying algorithms to learn from data and make automated predictions.
- ¥A machine learning engineer is essentially a software engineer who is skilled enough in data science to deploy advanced data science models within the applications they build.
- ¥This person doesn't need to know as much data science as a data scientist but should know much more about computer science and software development than a typical data scientist.

Grasping the Differences among Data Approaches

| Defining data engineering

- ¥Data engineering is dedicated to build and maintain data systems for overcoming data processing bottlenecks and data handling problems that arise from managing the high volume, velocity, and variety of big data.
- ¥Data engineers often have experience working with real-time processing frameworks and massively parallel processing (MPP) platforms, as well as with RDBMSs. They generally code in Java, C++, Scala, or Python.
- ¥They know how to deploy Hadoop MapReduce or Spark to handle, process, and refine big data.

Grasping the Differences among Data Approaches

Comparing machine learning engineers, data scientists, and data engineers

- ¥The roles of data scientist, machine learning engineer, and data engineer are frequently conflated by hiring managers.
- ¥Data scientists are sometimes stuck having to learn to do the job of a data engineer, and vice versa.
- ¥To summarize, hire (i) a data engineer to store, migrate, and process your data; (ii) a data scientist to make sense of it for you; (iii) and a machine learning engineer to bring your machine learning models into production.



1. Storing and Processing Data for Data Science



Storing and Processing Data for Data Science

| Storing data and doing data science directly in the cloud

- ¥ Storing data in a cloud environment offers serious business advantages such as faster time-to-market, Enhanced flexibility, and security.
- ¥ A lot of different technologies have emerged after the cloud computing revolution, many of which are of interest to those trying to leverage big data. The following examine a few of these new technologies.

Storing and Processing Data for Data Science

Storing data and doing data science directly in the cloud

Using serverless computing to execute data science:

- ! Serverless computing refers to computing that's executed in a cloud environment rather than on your desktop or on-premise at your company.
- ! Serverless computing decreases the downtime that data scientists spend in preparing data and infrastructure for their predictive models.
- ! With serverless computing, your data science model runs directly within its container. Your cloud service provider handles all the adjustments that need to be made to the infrastructure to support your functions.

Storing and Processing Data for Data Science

■ Storing data and doing data science directly in the cloud

¥ Containerizing predictive applications within Kubernetes

- ! Kubernetes is an open-source software suite that manages, orchestrates, and coordinates the deployment, scaling, and management of containerized applications across clusters of worker nodes.
- ! One particularly attractive feature about Kubernetes is that you can run it on data that sits in on-premise clusters, in the cloud, or in a hybrid cloud environment.

Storing and Processing Data for Data Science

■ Storing data and doing data science directly in the cloud

¥ Sizing up popular cloud-warehouse solutions

! You have several products to choose from when it comes to cloud-warehouse solutions. The following list looks at the most popular options:

¥ Amazon Redshift

¥ Snowflake

¥ GoogleBigQuery

Storing and Processing Data for Data Science

Storing data and doing data science directly in the cloud

Introducing NoSQL databases

- ! A traditional RDBMS isn't equipped to handle big data demands.
- ! It's designed to handle only relational datasets constructed of data stored in clean rows and columns and thus is capable of being queried via SQL.
- ! RDBMSs are incapable of handling unstructured and semistructured data.
- ! RDBMSs simply lack capabilities needed for meeting big data volume and velocity requirements.

Storing and Processing Data for Data Science

Storing data and doing data science directly in the cloud

Introducing NoSQL databases

- ! This is where NoSQL comes in.
- ! Its databases are nonrelational, distributed database systems that were designed to rise to the challenges involved in storing and processing big data. They can be run on-premise or in a cloud environment.
- ! NoSQL systems facilitate non-SQL data querying of nonrelational or schema-free, semistructured and unstructured data.

Storing and Processing Data for Data Science

Storing big data on-premise

- ✖ Although cloud storage and cloud processing of big data is widely accepted as safe, reliable, and cost-effective, companies have a multitude of reasons for using on-premise solutions instead.
- ✖ Reminiscing about Hadoop To work around the limitations of relational systems, data engineers originally turned to the Hadoop data processing platform to boil down big data into smaller datasets that are more manageable for data scientists to analyze.

Storing and Processing Data for Data Science

Storing big data on-premise

- ✖ When people refer to Hadoop, they're generally referring to an on-premise Hadoop storage environment that includes the HDFS (for data storage), MapReduce (for bulk data processing), Spark (for real-time data processing), and YARN (for resource management).
- ✖ Incorporating MapReduce, the HDFS, and YARN :
MapReduce is a parallel distributed processing framework that can process tremendous volumes of data batch. MapReduce works by converting raw data down to sets of tuples and then combining and reducing those tuples into smaller sets of tuples.

Storing and Processing Data for Data Science

Storing big data on-premise

- ¥ Storing data on the Hadoop distributed file system (HDFS):
The HDFS uses clusters of commodity hardware for storing data. The HDFS is characterized by these three key features: HDS blocks, Redundancy, Fault-tolerance.
- ¥ Putting it all together on the Hadoop platform: The Hadoop platform was designed for large-scale data processing, storage, and management. This open-source platform is generally composed of the HDFS, MapReduce, Spark, and YARN (a resource manager) all working together.

Storing and Processing Data for Data Science

Storing big data on-premise

Introducing massively parallel processing (MPP) platforms : Massively parallel processing (MPP) platforms can be used instead of MapReduce as an alternative approach for distributed data processing. MPP runs parallel computing tasks on costly custom hardware, whereas MapReduce runs them on inexpensive commodity servers.

Processing big data in real-time

¥A real-time processing framework is a framework that processes data in real-time (or near-real-time) as the data streams and flows into the system.

¥Real-time processing frameworks do one of the following:

- ! Increase the overall time efficiency of the system.
- ! Deploy innovative querying methods to facilitate the real-time querying of big data.

Storing and Processing Data for Data Science

| Processing big data in real-time

- ¥ In-memory refers to processing data within the computer's memory, without actually reading and writing its computational results onto the disk. In-memory computing provides results a lot faster but cannot process much data per processing interval.
- ¥ Regardless of the industry in which you work, if your business is impacted by real-time data streams that are generated by humans, machines, or sensors, a real-time processing framework would be helpful to you in optimizing and generating value for your organization.





Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming
Module 4:
Machine Learning Means É Using a Machine to
Learn from Data



Contents

1. Defining Machine Learning and Its Processes
2. Considering Learning Styles
3. Seeing What You Can Do



Weekly Learning Outcomes

1. Grasping the machine learning process
2. Exploring machine learning styles and algorithms
3. Overviewing algorithms, deep learning, and Apache Spark



Required Reading

1. Chapter 3. Machine Learning Means É Using a Machine to Learn from Data (Lillian Pierson, Data Science, 3rd Edition, 2021)

Videos

- ¥ Machine Learning Basics | What Is Machine Learning? | Introduction To Machine Learning

<https://youtu.be/ukzFI9rgwfU>

- ¥ Supervised vs Unsupervised vs Reinforcement Learning

<https://youtu.be/1FZ0A1QCMWc>



1. Defining Machine Learning and Its Processes



Defining Machine Learning and Its Processes

- ✖ Machine learning is the practice of applying algorithmic models to data over and over again so that your computer discovers hidden patterns or trends that you can use to make predictions. It's also called algorithmic learning.
- ✖ Machine learning has a vast and ever-expanding assortment of use cases, including Real-time Internet advertising, Spam filtering, Recommendation engines, Natural language processing and sentiment analysis, Automatic facial recognition, etc.

Defining Machine Learning and Its Processes

| Walking through the steps of the machine learning process

¥Three main steps are involved in machine learning: setup, learning, and application.

¥Setup involves acquiring data, preprocessing it, selecting the most appropriate variables for the task at hand (called feature selection), and breaking the data into training and test datasets. You use the training data to train the model, and the test data to test the accuracy of the model's predictions.

Defining Machine Learning and Its Processes

| Walking through the steps of the machine learning process

- ¥The learning step involves model experimentation, training, building, and testing.
- ¥The application step involves model deployment and prediction.
- ¥Here's a rule of thumb for breaking data into test and training sets: Apply random sampling to two-thirds of the original dataset in order to use that sample to train the model. Use the remaining one-third of the data as test data, for evaluating the model's predictions.

Defining Machine Learning and Its Processes

| Becoming familiar with machine learning terms

¥ Because machine learning is a branch of both traditional statistics and computer science, it has adopted terms from both fields and added a few of its own. Here is what you need to know:

- ! Instance: The same as a row (in a data table), an observation (in statistics), and a data point. Machine learning practitioners are also known to call an instance case.
- ! Feature: The same as a column or field (in a data table) and a variable (in statistics). In regression methods, a feature is also called an independent variable (IV).

Defining Machine Learning and Its Processes

| Becoming familiar with machine learning terms

! Target variable: The same as predictor or dependent variable (DV) in statistics.

¥ Although machine learning is often referred to in context of data science and artificial intelligence, these terms are all separate and distinct. Machine learning is a practice within data science, but there is more to data science than just machine learning



1. Considering Learning Styles



Considering Learning Styles

Machine learning can be applied in three main styles: supervised, unsupervised, and semisupervised. Supervised and unsupervised methods are behind most modern machine learning applications, and semisupervised learning is an up and coming star.

Considering Learning Styles

| Learning with supervised algorithms

- ¥Supervised learning algorithms require that input data has labeled features.
- ¥These algorithms learn from known features of that data to produce an output model that successfully predicts labels for new incoming, unlabeled data points.
- ¥You use supervised learning when you have a labeled dataset composed of historical values that are good predictors of future events.
- ¥Logistic regression is a type of supervised learning

Considering Learning Styles

| Learning with unsupervised algorithms

- ¥ Unsupervised learning algorithms accept unlabeled data and attempt to group observations into categories based on underlying similarities in input features.
- ¥ Principal component analysis, k-means clustering, and singular value decomposition are all examples of unsupervised machine learning algorithms.

Considering Learning Styles

| Learning with reinforcement

- ¥ Reinforcement learning is a behavior-based learning model.
- ¥ It's based on a mechanic like how humans and animals learn.
- ¥ The model is given "rewards" based on how it behaves, and it subsequently learns to maximize the sum of its rewards by adapting the decisions it makes to earn as many rewards as possible.

1. Seeing What You Can Do

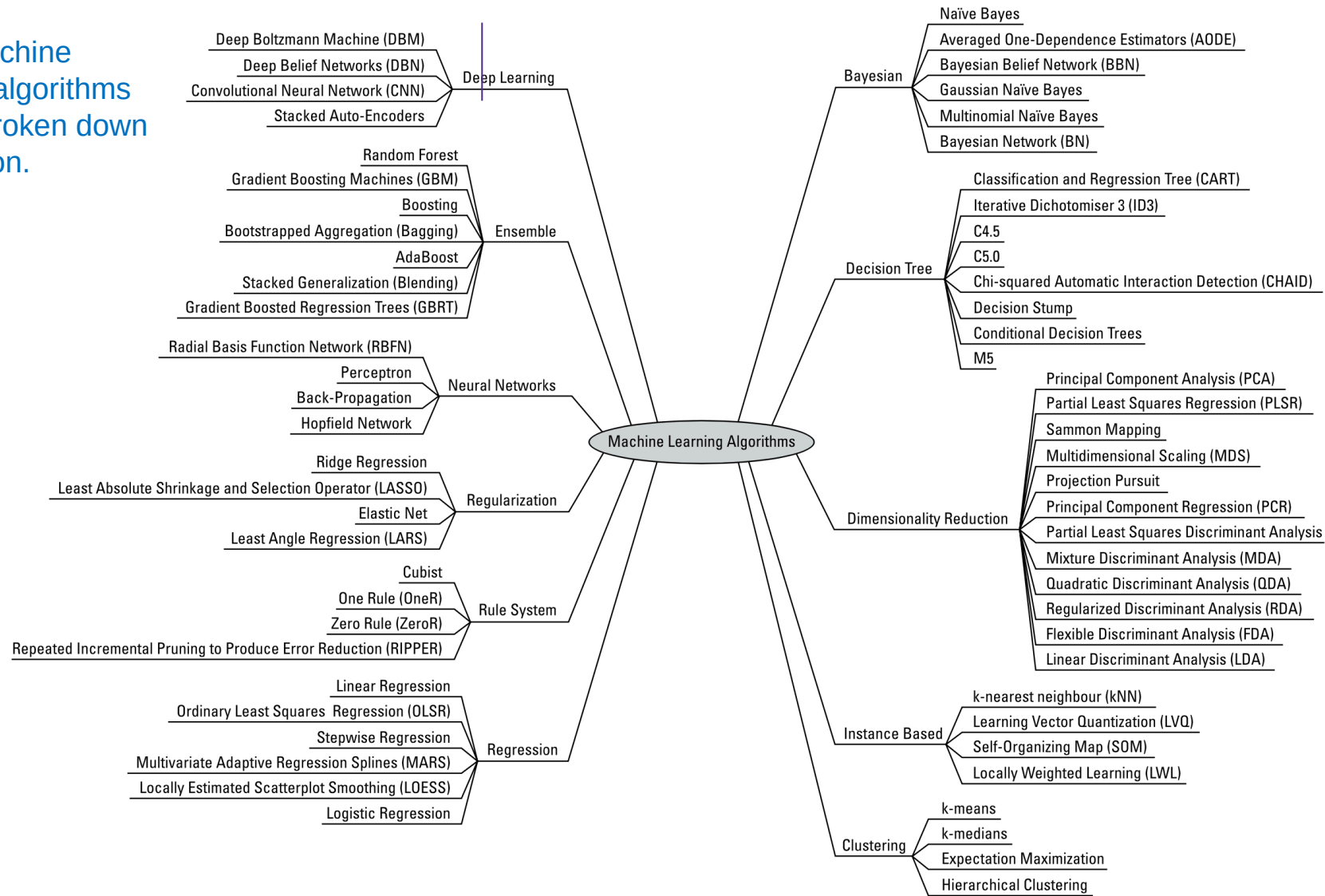


Seeing What You Can Do

| Selecting algorithms based on function

When you need to choose a class of machine learning algorithms, it's helpful to consider each model class based on its functionality. For the most part, algorithmic functionality falls into the categories shown in the following Figure.

FIGURE 3-3: Machine learning algorithms can be broken down by function.



Seeing What You Can Do

Selecting algorithms based on function

¥Example of Deep learning algorithm: If you use Gmail, you must be enjoying its autoreply functionality. You know N the three 1-line messages from which you can choose an autoreply to a message someone sent you? Well, this autoreply functionality within Gmail is called Smart Reply, and it is built on deep learning algorithms.

¥Another innovation built on deep learning is Facebook DeepFace, the Facebook feature that automatically recognizes and suggests tags for the people who appear in your Facebook photos.

Seeing What You Can Do

| Using Spark to generate real-time big data analytics

- ¥Apache Spark is an in-memory distributed computing application that you can use to deploy machine learning algorithms on big data sources in near-real-time to generate analytics from streaming big data sources.
- ¥In-memory refers to processing data within the computer's memory, without reading and writing its computational results onto the disk. In-memory computing provides its results a lot faster but cannot process much data per processing interval.





Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming

Module 5: Probability, and Statistical Modeling



Contents

1. Exploring Probability and Inferential Statistics
2. Quantifying Correlation
3. Reducing Data Dimensionality with Linear Algebra
4. Modeling Decisions with Multiple Criteria Decision-Making
5. Introducing Regression Methods
6. Detecting Outliers
7. Introducing Time Series Analysis



Weekly Learning Outcomes

1. Introducing the core basics of statistical probability
2. Quantifying correlation
3. Reducing dataset dimensionality
4. Building decision models with multiple criteria decision-making
5. Diving into regression methods
6. Detecting outliers
7. Talking about time series analysis



Required Reading

1. Chapter 4. Math, Probability, and Statistical Modeling
(Lillian Pierson, Data Science, 3rd Edition, 2021)

Videos

- ¥ Math Antics - Basic Probability
<https://youtu.be/KzfWUEJjG18>
- ¥ Statistics - How to find outliers
<https://youtu.be/9aDHbRb4Bf8>



Introduction



Introduction |

- ¥In data science, the need for the math and statistic methods is simply a fact of life.
- ¥Data scientists use statistics, math, coding, and strong communication skills to help them discover, understand, and communicate data insights that lie within raw datasets related to their field of expertise.
- ¥In this chapter, we introduce the basic ideas behind probability, correlation analysis, dimensionality reduction, decision modeling, regression analysis, outlier detection, and time series analysis.

1. Exploring Probability and Inferential Statistics



Exploring Probability and Inferential Statistics

¥A statistic is a result that's derived from performing a mathematical operation on numerical data. In general, you use statistics in decision-making. Statistics come in two flavors:

! Descriptive: Descriptive statistics provide a description that illuminates some characteristic of a numerical dataset, including dataset distribution, central tendency (such as mean, min, or max), and dispersion (as in standard deviation and variance). Descriptive statistics are not meant to illustrate any causal claims.

Descriptive statistics can highlight relationships between X and Y, but they do not assume that X causes Y.

Exploring Probability and Inferential Statistics

! Inferential: inferential statistics carve out a smaller section of the dataset and attempt to deduce significant information about the larger dataset.

Unlike descriptive statistics, inferential methods, such as regression analysis, DO try to predict by studying causation.

Use this type of statistics to derive information about a real-world measure in which you're interested.

Exploring Probability and Inferential Statistics

- ✖ It's true that descriptive statistics describe the characteristics of a numerical dataset, but that doesn't tell you why you should care. In fact, most data scientists are interested in descriptive statistics only because of what they reveal about the real-world measures they describe.
- ✖ To better understand this concept, imagine that a business owner wants to estimate the upcoming quarter's profits. The owner might take an average of the past few quarters' profits use as an estimate of how much profit they'll make during the next quarter.

Exploring Probability and Inferential Statistics

- ¥ But if the previous quarters' profits varied widely, a descriptive statistic that estimated the variation of this predicted profit value (the amount by which this dollar estimate could differ from the actual profits earned) would indicate just how far the predicted value could be from the actual one.
- ¥ You can use descriptive statistics in many ways Ñ to detect outliers, for example, or to plan for feature preprocessing requirements or to quickly identify which features you may want Ñ or not want Ñ to use in an analysis.

Exploring Probability and Inferential Statistics

¥Inferential statistics do this by providing information about a small data selection, so you can use this information to infer something about the larger dataset from which it was taken. In statistics, this smaller data selection is known as a sample, and the larger, complete dataset from which the sample is taken is called the population.

¥You can also use inferential statistics in situations where you simply can't afford to collect data for the entire population.

Exploring Probability and Inferential Statistics

- ¥ At other times, you may find yourself in situations where complete information for the population isn't available. In these cases, you can use inferential statistics to estimate values for the missing data based on what you learn from analyzing the data that is available.
- ¥ For an inference to be valid, you must select your sample carefully so that you form a true representation of the population. Even if your sample is representative, the numbers in the sample dataset will always exhibit some noise – random variation.

Exploring Probability and Inferential Statistics

| Probability distributions

- ¥If your goal is to model an unpredictable variable so that you can make data-informed decisions based on what you know about its probability in a population, you can use random variables and probability distributions to do this.
- ¥When considering the probability of an event, you must know what other events are possible. Always define the set of event as mutually exclusive $\cap$ only one can occur at a time. (Think of the six possible results of rolling a die.

Exploring Probability and Inferential Statistics

Probability distributions

Probability has these two important characteristics:

- ! The probability of any single event never goes below 0.0 or exceeds 1.0.
- ! The probability of all events always sums to exactly 1.0.

Probability distribution is classified per these two types:

- ! Discrete: A random variable where values can be counted by groupings.
- ! Continuous: A random variable that assigns probabilities to a range of values.

Exploring Probability and Inferential Statistics

| Probability distributions

- ¥To understand discrete and continuous distribution, think of two variables from a dataset describing cars.
- ¥A "color" variable would have a discrete distribution because cars have only a limited range of colors (black, red, or blue, for example). The observations would be countable per the color grouping.
- ¥A variable describing cars' miles per gallon, would have a continuous distribution because each car could have its own, separate value for miles per gallon that it gets on average.

Exploring Probability and Inferential Statistics

| Probability distributions

- ✖ Normal distributions (numeric continuous): Represented graphically by a symmetric bell-shaped curve, these distributions model phenomena that tend toward some most-likely observation (at the top of the bell in the bell curve); observations at the two extremes are less likely.
- ✖ Binomial distributions (numeric discrete): These distributions model the number of successes that can occur in a certain number of attempts when only two outcomes are possible. Binary variables $\tilde{N}$ variables that assume only one of two values $\tilde{N}$ have a binomial distribution.

Exploring Probability and Inferential Statistics

| Probability distributions

¥Categorical distributions (non-numeric): These represent either nonnumeric categorical variables or ordinal variables (an ordered categorical variable, for example the level of service offered by most airlines is ordinal because they offer first class business class, and economy class seats).

Exploring Probability and Inferential Statistics

Conditional probability with Naïve Bayes

- ✖ You can use the Naïve Bayes machine learning method, which was borrowed straight from the statistics field, to predict the likelihood that an event will occur, given evidence defined in your data features Ñ something called conditional probability.
- ✖ Naïve Bayes, which is based on classification and regression, especially useful if you need to classify text data. Example: Spam classification.



1. Quantifying Correlation



Quantifying Correlation

- Correlation is quantified per the value of a variable called r , which ranges between -1 and 1 .
- The closer the value is to 1 or -1 , the more correlation there is between two variables.
- If two variables have a value that's close to 0 , it could indicate that they're independent variables.

Quantifying Correlation

Calculating correlation with Pearson's r

✖ If you want to uncover dependent relationships between continuous variables in a dataset, you'd use statistics to estimate their correlation.

✖ The simplest form of correlation analysis is Pearson correlation, which assumes that:

- ! Your data is normally distributed.
- ! You have continuous, numeric variables.
- ! Your variables are linearly related.

Quantifying Correlation

Ranking variable-pairs using Spearman's rank correlation

¥The Spearman's rank correlation is a popular test for determining correlation between ordinal variables.

¥By applying Spearman's rank correlation, you're converting numeric variable-pairs into ranks by calculating the strength of the relationship between variables and then ranking them per their correlation.

¥The Spearman's rank correlation assumes that

- ! Your variables are ordinal.
- ! Your variables are related nonlinearly.



1. Reducing Data Dimensionality with Linear Algebra



Reducing Data Dimensionality with Linear Algebra

- ¥ Any intermediate-level data scientist should have a good understanding of linear algebra and how to do math using matrices.
- ¥ Array and matrix objects are the primary data structure in analytical computing.
- ¥ You need them in order to perform mathematical and statistical operations on large and multidimensional datasets with many different features to be tracked simultaneously.

Reducing Data Dimensionality with Linear Algebra

| Decomposing data to reduce dimensionality

- ✖ Using a linear algebra method called Singular Value Decomposition (SVD), you can reduce the dimensionality of your dataset.
- ✖ Dimension reduction algorithms are ideal options if you need to compress your dataset while also removing redundant information and noise.
- ✖ In data science, SVD is applied to analyze principal components from with large, noisy, sparse data sets Ñ an approach machine learning called Principal Component Analysis (PCA).

Reducing Data Dimensionality with Linear Algebra

| Reducing dimensionality with factor analysis

- ¥ Factor analysis is along the same lines as SVD in that it's a method you can use for filtering out redundant information and noise from your data.
- ¥ A variable's variability measures how much variance it has around its mean. The greater a variable's variance, the more information that variable contains.
- ¥ When you find shared variance in your dataset, that means information redundancy is at play. You can use factor analysis or principal component analysis to clear your data of this information redundancy.

Reducing Data Dimensionality with Linear Algebra

| Reducing dimensionality with factor analysis

Factor analysis makes the following assumptions:

- ! Your features are metric
- ! Your features should be continuous or ordinal.
- ! You have more than 100 observations in your dataset and at least 5 observations per feature.
- ! Your sample is homogenous.
- ! There is $r > 0.3$ correlation between the features in your dataset.

Reducing Data Dimensionality with Linear Algebra

| Decreasing dimensionality and removing outliers with PCA

- ¥ Principal component analysis (PCA) is another dimensionality reduction technique that's closely related to SVD.
- ¥ This unsupervised statistical method finds relationships between features in your dataset and then transforms and reduces them to a set of non-information redundant principal components.
- ¥ You can then use those reduced components as input for your machine learning algorithms to make predictions based on a compressed representation of your data.

Reducing Data Dimensionality with Linear Algebra

Decreasing dimensionality and removing outliers with PCA

Although PCA is like factor analysis, they have two major differences:

- ! One difference is that PCA does not regress to find some underlying cause of shared variance, but instead decomposes dataset to succinctly represent its most important information a reduced number of features.
- ! The other key difference is that, with PCA, the first time you run the model, you don't specify the number of components discovered in the dataset. You let the initial model results tell you how many components to keep, and then you rerun the analysis to extract those features.



1. Modeling Decisions with Multiple Criteria Decision-Making



Modeling Decisions with Multiple Criteria Decision-Making

| Turning to traditional MCDM

- ¥We're often forced to make decisions where several different criteria come into play, and it often seems unclear which criterion should have priority.
- ¥Anywhere you have two or more criteria on which you need to base your decision, you can use MCDM methods to help you evaluate alternatives.

Modeling Decisions with Multiple Criteria Decision-Making |

| Turning to traditional MCDM

¥To use multiple criteria decision-making, the following two assumptions must be satisfied:

- ! Multiple criteria evaluation: You must have more than one criterion to optimize.
- ! Zero-sum system: Optimizing with respect to one criterion must come at the sacrifice of at least one other criterion.

Modeling Decisions with Multiple Criteria Decision-Making |

| Focusing on fuzzy MCDM

- ¥ If you prefer to evaluate suitability within a range, rather than use binary membership terms of 0 or 1, you can use fuzzy multiple criteria decision-making (FMCDM) to do that.
- ¥ The term fuzzy refers to the fact that the criteria being used to evaluate alternatives offer a range of acceptability - instead of the binary, crisp set criteria associated with traditional MCDM.

1. Introducing Regression Methods



Introducing Regression Methods

- Machine learning algorithms of the regression variety were adopted from the statistics field in order to provide data scientists with a set of methods for describing and quantifying the relationships between variables in a dataset.
- Use regression techniques if you want to determine the strength of correlation between variables in your data.
- As for using regression to predict future values from historical values, feel free to do it, but be careful: Regression methods assume a cause-and-effect relationship between variables.

Introducing Regression Methods

Linear regression

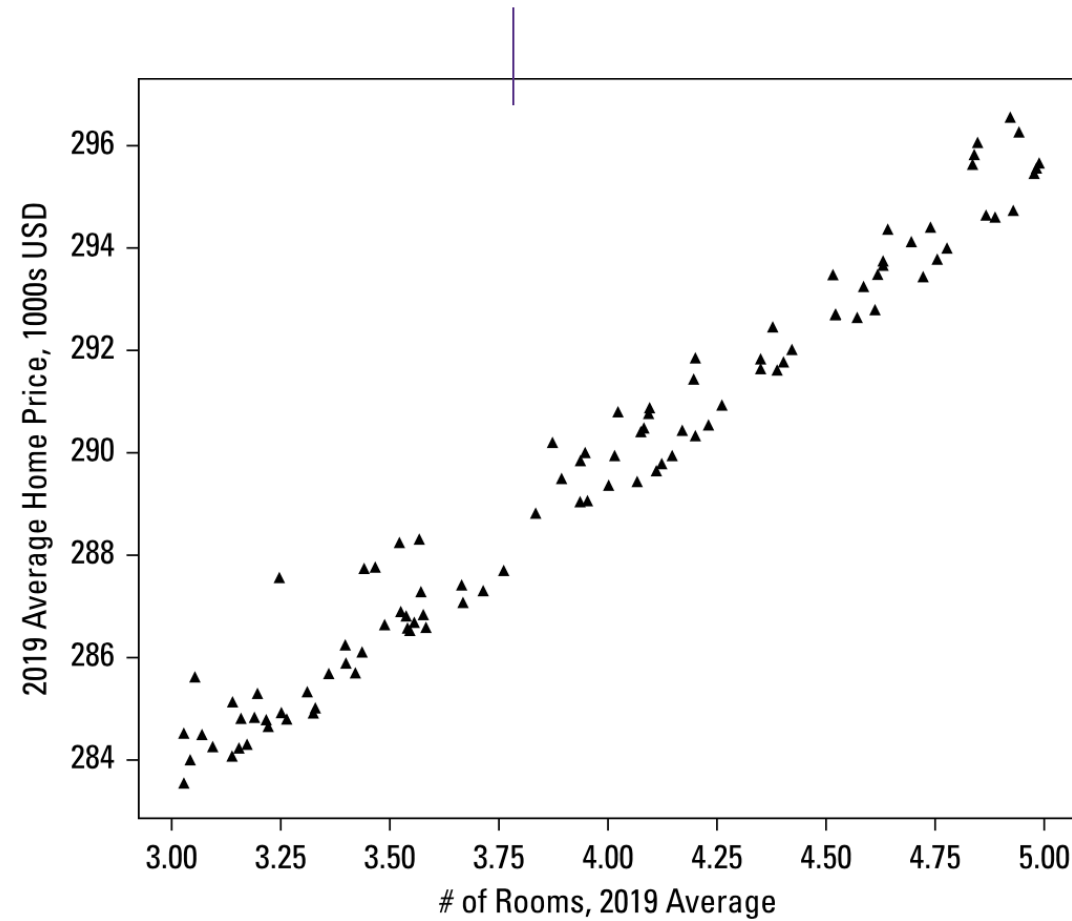
- ✖ Linear regression is a machine learning method you can use to describe and quantify the relationship between your target variable, and the dataset features you've chosen to use as predictor variables.
- ✖ When you use just one variable as your predictor, linear regression is as simple as the middle school algebra formula $y=mx+b$.
- ✖ Linear regression limitations:

Introducing Regression Methods

Linear regression

- ! Linear regression works with only numerical variables, not categorical ones.
- ! If your dataset has missing values, it will cause problems. Be sure to address your missing values before attempting to build a linear regression model.
- ! If your data has outliers present, your model will produce inaccurate results.
- ! The linear regression model assumes that a linear relationship exists between dataset features and the target variable.
- ! The linear regression model assumes that all features are independent of each other.
- ! Prediction errors, or residuals, should be normally distributed.

FIGURE 4-6:
Linear regression
used to predict
home prices
based on the
number of rooms
in a house.



Introducing Regression Methods

Logistic regression

- ¥ Logistic regression is a machine learning method you can use to estimate values for a categorical target variable based on your selected features.
- ¥ Your target variable should be numeric and should contain values that describe the target's class.
- ¥ One cool aspect of logistic regression is that, in addition to predicting the class of observations in your target variable, it indicates the probability for each of its estimates.

Introducing Regression Methods

Ordinary least squares (OLS) regression methods

- Ordinary least squares (OLS) is a statistical method that fits a linear regression line to a dataset.
- Use OLS if you want to construct a function that's a close approximation to your data.
- OLS is particularly useful for fitting a regression line to models containing more than one independent variable. In this way, you can use OLS to estimate the target from dataset features.



1. Detecting Outliers



Detecting Outliers

Analyzing extreme values

- ¥ Many statistical and machine learning approaches assume that your data has no outliers. Outlier removal is an important part of preparing your data for analysis.
- ¥ Outliers are data points with values that are significantly different from the majority of data points comprising a variable.
- ¥ You can use outlier detection to spot anomalies that represent fraud, equipment failure, or cybersecurity attacks.
- ¥ You can detect outliers using either a univariate or multivariate approach, as spelled out in the next two sections.

Detecting Outliers |

| Detecting outliers with univariate analysis

¥Univariate outlier detection is where you look at features in your dataset and inspect them individually for anomalous values.

¥You can choose from two simple methods for doing this:

- ! Tukey outlier labeling
- ! Tukeyboxplotting

Detecting Outliers |

| Detecting outliers with multivariate analysis

- ¥ Sometimes outliers show up only within combinations of data points from disparate variables. These outliers cause problem on machine learning algorithms, so it's important to detect and remove them.
- ¥ You can use multivariate analysis of outliers to do this.
- ¥ A multivariate approach to outlier detection involves considering two or more variables at a time and inspecting them together for outliers.

Detecting Outliers |

| Detecting outliers with multivariate analysis

¥You can use one of several methods, including:

- ! A scatter-plot matrix
- ! Boxplotting
- ! Density-based spatial clustering of applications with noise (DBScan)
- ! Principal component analysis



1. Introducing Time Series Analysis



Introducing Time Series Analysis



- ¥A time series is just a collection of data on attribute values over time.
- ¥Time series analysis is performed to predict future instances of the measure based on the past observational data.
- ¥To forecast or predict future values from data in your dataset, use time series techniques.

Introducing Time Series Analysis

Identifying patterns in time series

Time series exhibit specific patterns. Take a look at following Figure to gain a better understanding of what these patterns are all about.

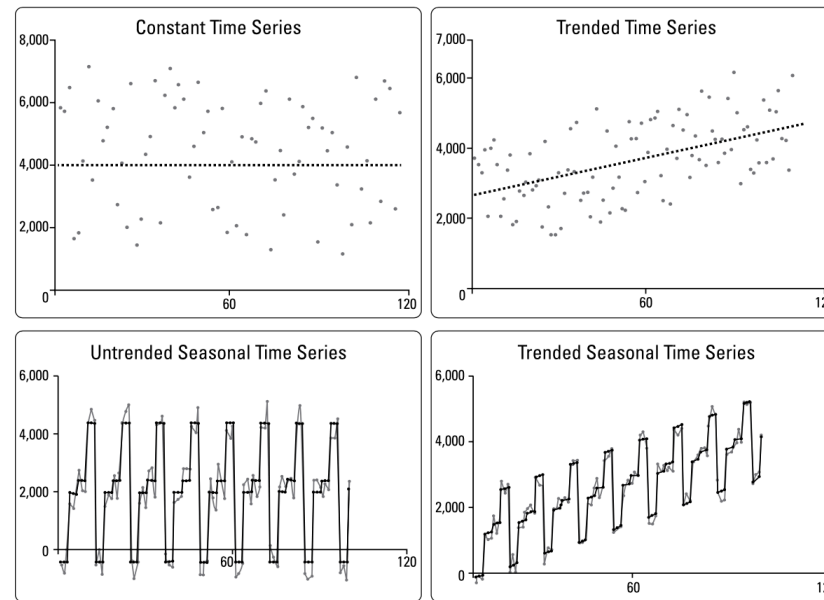


FIGURE 4-9:
A comparison
of patterns
exhibited by
time series.

Introducing Time Series Analysis

Modeling univariate time series data patterns in time series

- ¥ Similar to how multivariate analysis is the analysis of relationships between multiple variables, univariate analysis is the quantitative analysis of only one variable at a time.
- ¥ When you model univariate time series, you're modeling time series changes that represent changes in a single variable over time.





Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming

Module 6:

¥ Part 1: Ten Phenomenal Resources for Open Data

¥ Part 2: Starting with Python



Contents

Part 1: Ten Phenomenal Resources for Open Data

1. Exploring Data Worldwide
2. Discovering Saudi Open Data Portal

Part 2: Starting with Python

3. Why Python Is Hot
4. Choosing the Right Python
5. Tools for Success
6. Writing Python in VS Code
7. Using Jupyter Notebook for Coding



Weekly Learning Outcomes

1. Learning about several resources for open Data
2. Discovering Saudi Open Data Portal <https://data.gov.sa/en>
3. Discovering Python
4. Writing Python Code



Required Reading

1. Chapter 19. Ten Phenomenal Resources for Open Data (Lillian Pierson, Data Science, 3rd Edition, 2021)
2. Book 1, Chapter 1. Starting with Python (John S. Kovic and Alan Simpson, Python All-in-One, 2nd edition 2021)

Videos

- ¥ Best Data Sources for your Next Analytic Projects | FREE Open-Source Datasets
<https://youtu.be/z3DsLSmkqOU>
- ¥ !"#&%&)*+\$.-.* /0123456&718
<https://youtu.be/8h7hWls5Nh4>



Part 1: Ten Phenomenal Resources for Open Data



Introduction



Introduction |

- ¥ Think of open data as data that has been made publicly available and is permitted to be used, reused, built on, and shared with others.
- ¥ Maybe you've heard of open-source software, open hardware, open-content creative work, open access to scientific journals, and open science. Along with open data, they're all part of the aptly named open movement.
- ¥ The distinguishing feature of open licenses is that they have copyleft instead of copyright.

Introduction |

- ¥Be aware that sometimes work that's labeled as open may not fit the accepted definition. You're responsible for checking the licensing rights and restrictions of the open data you use.
- ¥As part of more recent open government initiatives, governments around the world began releasing open government data.
- ¥This data can be used by volunteer analysts and programmers who work collaboratively to build open-source solutions that use open data to solve social problems.
- ¥The open government movement promotes government transparency and accountability.



1. Exploring Data Worldwide



Exploring Data Worldwide

| Digging Through data.gov

- ¥The Data.gov program ([at www.data.gov](http://www.data.gov)) provide open access to nonclassified US government data.
- ¥By mid-2014, you could search for over 100,000 datasets by using the Data.gov search. The website is an unparalleled resource for the following indicators: Economic, Environmental, STEM industry, Quality of life and Legal.
- ¥You can also find over 60 open-source application programming interfaces (APIs) available on the platform. You can use these APIs to create tools and apps that pull data from government departments listed in the Data.gov data catalog.

Checking Out Canada Open Data

¶ If you look at the Canada Open Data website (<http://open.canada.ca>) you can find over 200,000 datasets. Among the 25 most popular offerings on the site are datasets cover the following indicators:

- ! Environmental: like natural disasters and fuel consumption ratings
- ! Citizenship: Permanent resident applications, permanent resident counts, foreign student entry counts, and other items can be found.
- ! Quality of life: like cost of living trends, automobile collision statistics, and disease surveillance, for example.

Exploring Data Worldwide

| Diving into data.gov.uk

- ¥The United Kingdom got off to a late start in the open government movement. <http://data.gov.uk> was started in 2010, and by mid-2014, only about 20,000 datasets were yet available.
- ¥The website represents a useful source for data on the following indicators: Environmental, Government spending, Societal, Health, Education and Business and economic.

Exploring Data Worldwide

| Checking Out US Census Bureau Data

¥The demo- graphics data provided by the US Census Bureau (www.census.gov) can be extremely helpful if you're doing marketing or advertising research and need to target your audience according to the following classification: Age, Average annual income, Household size, Gender or race, and Level of education.

Exploring Data Worldwide

Accessing NASA Data

- ¥ Since its inception in 1958, NASA has made public all its nonclassified project data. NASA datasets have been growing even faster with recent improvements in satellite and communication technology.
- ¥ In fact, NASA now generates 4 terabytes of new earth-science data per day, which is equivalent to over a million MP3 files.
- ¥ NASA's open data portal is called <http://data.nasa.gov>. This portal is a source of all kinds of wonderful data, including data about: Astronomy and space, Climate, Life sciences, Geology and Engineering.

Exploring Data Worldwide

| Wrangling World Bank Data

- ¥The World Bank is an international financial institution that provides loans to developing countries.
- ¥Data available to the public at the World Bank Open Data page (<http://data.worldbank.org>)
- ¥You can use the website to download entire datasets or simply view the data visualizations online. You can also use the World Bank's Open Data API to access what you need.

Exploring Data Worldwide

Wrangling World Bank Data

¥World Bank Open Data supplies data on the following indicators (and many, many more)

- ! Agriculture and rural development
- ! Economy and growth
- ! Environment
- ! Science and technology
- ! Financial sector
- ! Poverty income

| Getting to Know Knoema Data

¥The Knoema platform houses a staggering 500+ databases, in addition to its 150 million time series & 150 million collections of data on attribute values over time, in other words Knoema includes, but isn't limited to, all these data sources:

- ! Government data from industrial nations:
- ! National public data from developing nations:
- ! United Nations data:
- ! International organization data
- ! Corporate data from global corporations

Exploring Data Worldwide

| Queuing Up with Quandl Data

¥Quandl(www.quandl.com) is a Toronto-based website that aims to be a search engine for numeric data.

¥The Quandl database includes links to over 10 million datasets

¥Quandl links to 2.1 million UN datasets and many other sources, including datasets in the Open Financial Data Project, the central banks, real estate organizations, and well-known think tanks.

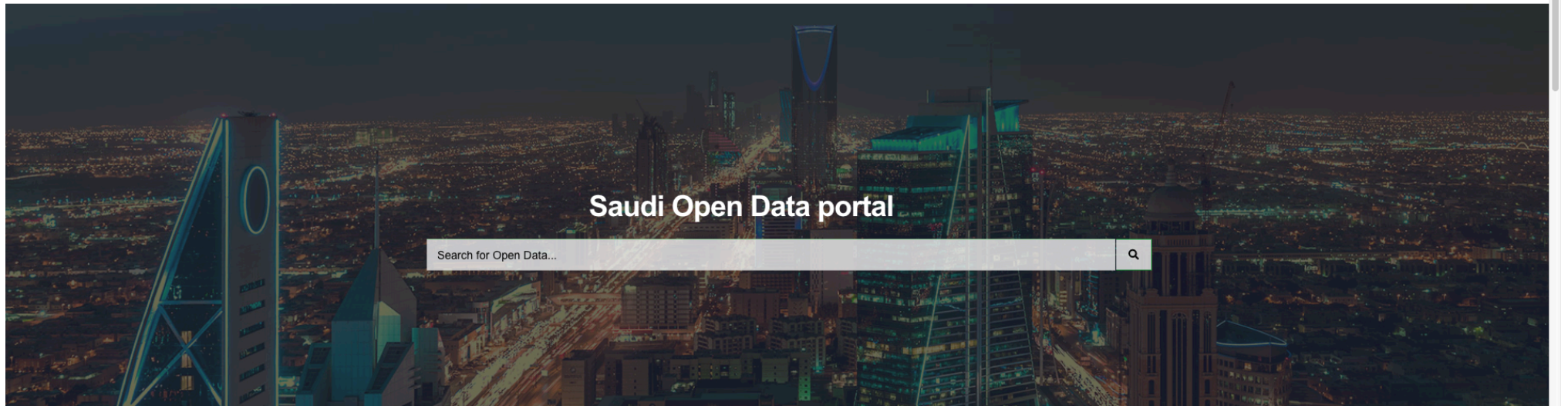
- ¥Exversion aims to provide the same collaborative functionality around data that GitHub provides around code. The Exversion platform offers version control functionality and hosting service for uploading and sharing your data.
- ¥All the data you upload to Exversion is public.
- ¥Exversion is extremely useful in the data-cleanup stage.

Mapping OpenStreetMap Spatial Data

- ¥OpenStreetMap (OSM) is an open, crowd-sourced alternative commercial mapping products such as Google Maps and ESRI ArcGIS Online.
- ¥To illustrate how a person can create data in OSM, imagine that someone links the GPS system on their mobile phone to the OSM application. Because of this authorization, OSM can automatically trace the routes of roads while the person travels. Later, this person (or another OSM user) can go to the OSM online platform to verify and label the routes.

1. Discovering Saudi Open Data Portal





Browse Groups

See All Groups



Health

Datasets 907



Agriculture and Fishing

Datasets 842



Social Services

Datasets 682



Trade (internal and external)

Datasets 648

Discovering Saudi Open Data Portal

- ¥The Open Data portal of Saudi Arabia is an important initiative for the country, as it aims to implement a public data hub and strategy to enable transparency, promote e-participation and inspire innovation.
- ¥The primary role of the portal is to publish datasets from ministries and government agencies in an open format, and make this data available to the public.

<https://data.gov.sa/en/aboutodp>

Discovering Saudi Open Data Portal

- ¥The platform enables the public to have central point of access to find, download and use datasets generated by the ministries and governmental entities in the country. Government Open Data helps bridge the gap between governments and citizens
- ¥The public benefits from the data provided in different ways, such as acquiring a better understanding of how to government agencies work, opening up the opportunities for people to evaluate the performance of various administrative institutions, giving citizens the chance to make informed decisions about government policies.

<https://data.gov.sa/en/aboutodp>

Discovering Saudi Open Data Portal

- ✚ It also allows using the data for research, reports, providing feedback, and developing web and smartphone applications and solutions based on government open data.
- ✚ With this Open Data portal the Saudi government pursues to expand its e-Government services portfolio by extending the growing efforts to individuals and private sectors, enhancing transparency and allowing people to show their creativity.

<https://data.gov.sa/en/aboutodp>

Discovering Saudi Open Data Portal

¥6,544 Datasets (August 2022)

¥Total publishers: 147

¥Use cases available

¥You may request Dataset

¥Open data license : The open data license provides freedom to users in distributing and producing works regarding the dataset, transforming and building upon them. The source should be mentioned in the prescribed manner in the license.

<https://data.gov.sa/sites/default/files/odp/Open%20Data%20License%201.02.pdf>

<https://data.gov.sa/en/aboutodp>



Lets discover the Saudi Open Data together

<https://data.gov.sa>



Part 2 : Starting with Python



1. Why Python Is Hot



Why Python Is Hot |

- ¥The Python language is becoming more and more popular, and in 2017 it became the most popular language in the world according to IEEE Spectrum.
- ¥Machine learning, robotics, artificial intelligence, and data science are the leading technologies today and for the future. Python is popular mainly because it already has lots of capabilities in these areas.

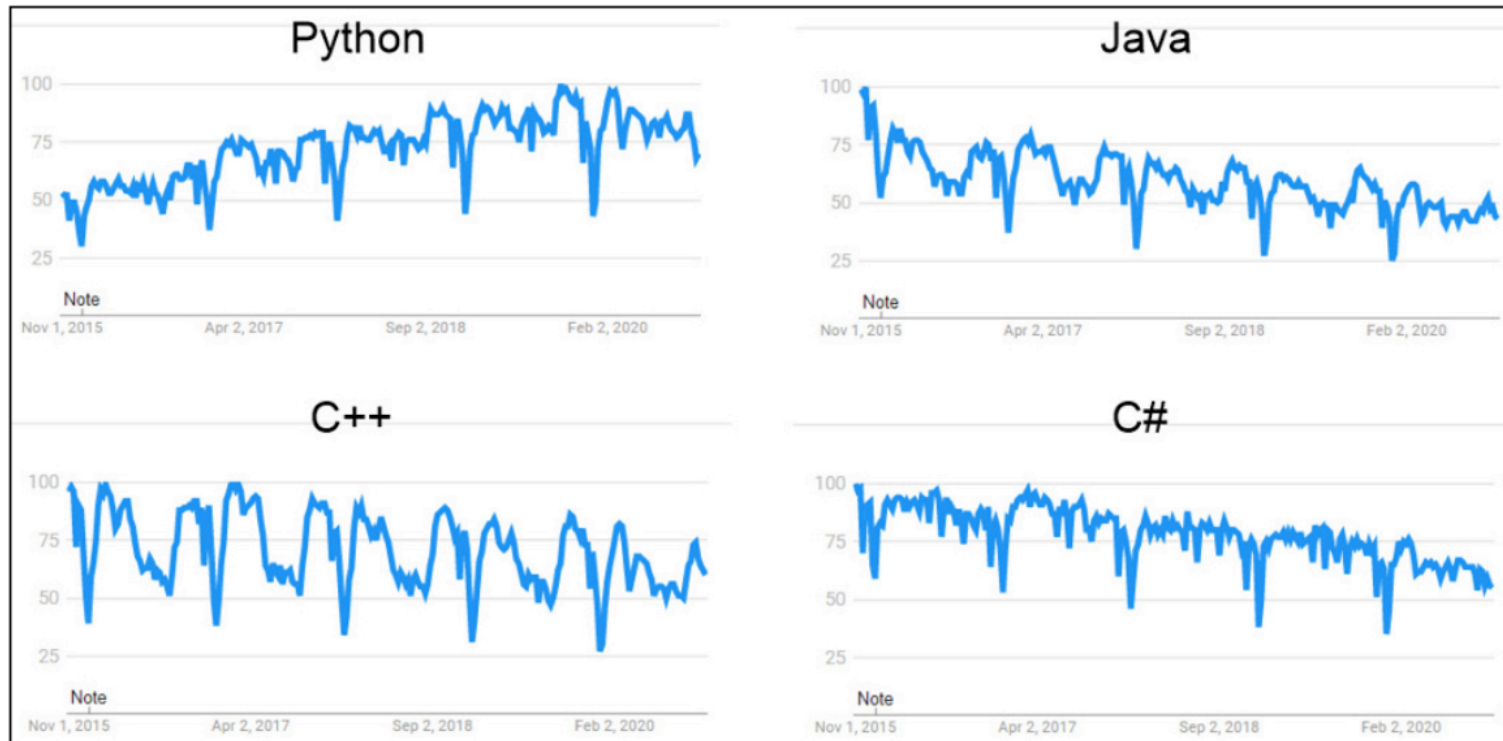
Why Python Is Hot |

¥The main reasons cited for Python's current popularity are:

- ! Python is relatively easy to learn.
- ! Everything you need to learn (and do) in Python is free.
- ! Python offers more ready-made tools for current hot technologies such as data science, machine learning, artificial intelligence, and robotics than most other languages.

¥Following figure shows Google search trends over the last five years. As you can see, Python has been gaining in popularity.

FIGURE 1-1:
Google search
trends for the last
five years or so.



1. Choosing the Right Python



Choosing the Right Python



¥This table shows the release dates of various Python versions

¥Go to python.org website to download Python, it will tell you the most current stable build (version). That's the one they recommend, and that's the one you should use.

Examples of Python Versions and Release Dates

Version	When Released
Python 3.9	October 2020
Python 3.8	October 2019
Python 3.7	June 2018
Python 3.6	December 2016
Python 3.5	September 2015
Python 3.4	March 2014
Python 3.3	September 2012
Python 3.2	February 2011
Python 3.1	June 2009
Python 3.0	December 2008
Python 2.7	July 2010
Python 2.6	October 2008
Python 2.0	October 2000
Python 1.6	September 2000
Python 1.5	February 1997
Python 1.0	January 1994



1. Tools for Success



Tools for Success

¥You'll need a good Python interpreter and editor. The editor lets you type the code, and the interpreter lets you run that code.

¥A code editor is an app that lets you type code. But you also need the Python interpreter.

¥Anaconda is a complete Python development environment with an intuitive and easily managed graphic user interface, as on a Mac or Windows or any phone or tablet.

¥Anaconda is often referred to as a data science platform because many of the packages that come with it are data science oriented. And it also comes with VS Code.

Installing Anaconda and VS Code

¥Go to www.anaconda.com/download

¥Click Download under the largest version number.

¥Follow any on screen instructions to download the free version

¥Installation: Choose whichever option makes sense to you.

¥When you come to a page where it asks if you want to install Microsoft VS Code (it may take quite a while), click Install Microsoft VS Code (or whatever option on your screen indicates that you want to install VS Code).

Tools for Success

Installing Anaconda and VS Code

Using Anaconda Navigator Anaconda Navigator lets you navigate around through different features of the app and choose what you want to run.

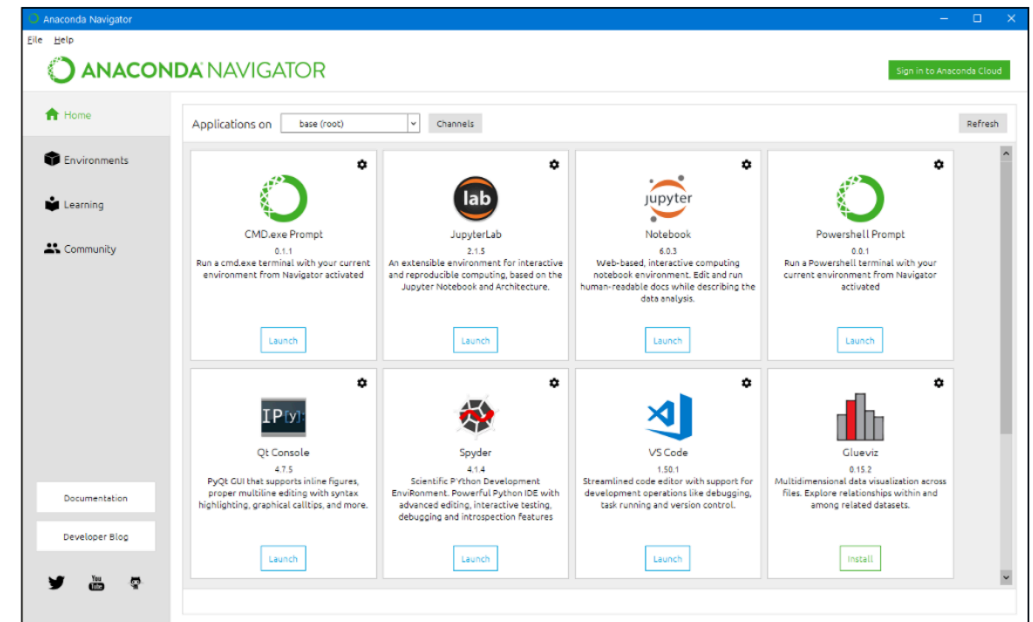


FIGURE 1-5:
Anaconda
Navigator
home page.

1. Writing Python in VS Code



Writing Python in VS Code

- ¥ We suggest that you open VS Code from Anaconda (Scroll down a little until you see the Launch button under VS Code, if necessary, and then click the Launch button)
- ¥ To use VS Code with Python and Anaconda, you need some VS Code extensions. You should already have them because they come with your Anaconda download. To verify that, click the Extensions icon in the left pane (it looks like a puzzle piece). You should see at least three extensions listed: Anaconda Extension Pack, Python, and YAML.

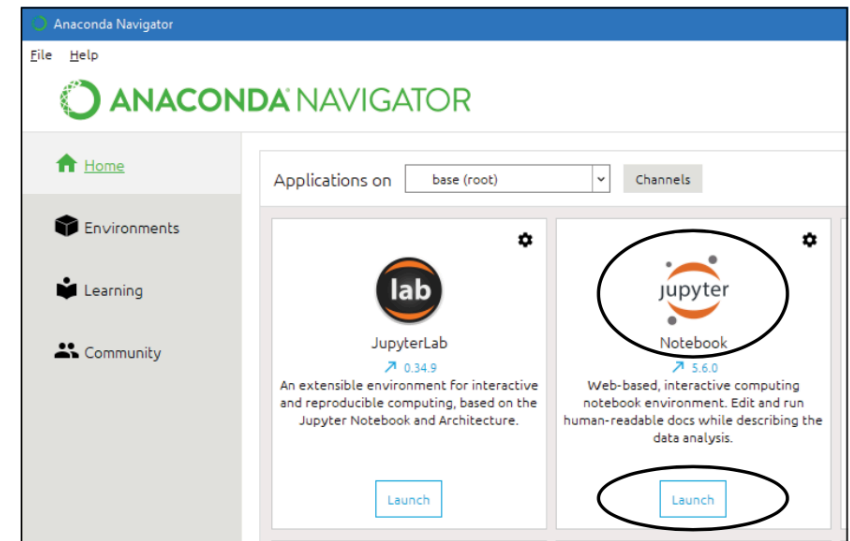
1. Using Jupyter Notebook for Coding



Using Jupyter Notebook for Coding

- ¥ Jupyter Notebook is another popular tool for writing Python code. It supports writing code in three popular languages: ~~Java~~ and Python and R.
- ¥ People often use Jupyter to share code on the Internet. It is free and comes with Anaconda.

FIGURE 1-11:
Launch Jupyter
Notebook from
Anaconda's
home page.





Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming

Module 7:

Interactive Mode, Getting Help, and Writing Apps



Contents

1. Using Python's Interactive Mode
2. Creating a Python Development Workspace
3. Creating a Folder for Your Python Code
4. Typing, Editing, and Debugging Python Code
5. Writing Code in a Jupyter Notebook



Weekly Learning Outcomes

1. Using interactive mode
2. Creating a development workspace
3. Creating a folder for your code
4. Typing, editing, and debugging code
5. Writing code in a Jupyter



Required Reading

1. Book 1, Chapter 2. Interactive Mode, Getting Help, and Writing Apps (John CShovicand Alan Simpson, Python All-in-One 2nd edition 2021)

Videos

- ¥ Jupyter Notebook Tutorial for Beginners with Python
<https://youtu.be/2WL-XTI2QYI>
- ¥ Python Tutorial - Introduction to DEBUGGING
<https://youtu.be/b7VbiZBg-dA>



Introduction



Using Python's Interactive Mode

- ✖ In this chapter, we take you briefly through the interactive, help, and code-editing features of VS Code and Jupyter Notebook to build on what you've learned so far.
- ✖ With Anaconda, the Terminal pane in VS Code is a great place to type Python code. So in this chapter that's where you'll start.

Using Python's Interactive Mode

| Opening Terminal

- ¥To use Python interactively with Anaconda, follow these steps:
- ¥Open Anaconda Navigator, and then open VS Code by clicking its Launch button on the Anaconda home page.
- ¥If you don't see the Terminal pane at the bottom of the VS Code window, choose View Terminal from the VS Code menu bar.
- ¥If the word Terminal isn't highlighted at the top of the pane, click Terminal

Using Python's Interactive Mode

| Opening Terminal



FIGURE 2-1:
The Terminal
pane in VS Code.

Depending on your Windows version and current configuration, you might see the following prompt instead, where xxx is your user name:

```
PS C:\Users\xxx>.
```

Using Python's Interactive Mode

| Getting your Python version

- ✖ At the operating system command prompt, type the following and press Enter to see what version of Python you're using. Note the space before the first hyphen, and no other spaces.
- ✖ You should see something like Python 3.x.x (where the x's are numbers representing the version of Python you're using).

```
python --version
```

Using Python's Interactive Mode

| Going into the Python Interpreter

¥When you're able to enter `python --version` and not get an error, you're ready to work with Python in VS Code. From there you can get into the Python interpreter by entering the command.

```
python
```

¥You should see some information about the Python version you're using and the `>` prompt, which represents the Python interpreter.

Using Python's Interactive Mode

| Entering commands

¥ Entering commands in the Python interpreter is the same as typing them anywhere else. You must type the command correctly, and then press Enter. If you spell something wrong in the command, you will likely see an error message.

| Using Python's built-in help

¥ You can type `help` as a command in the Python interpreter.

```
help()
```

¥ Seeing `help>` at the bottom of the window tells you that you're no longer in the Python interpreter (which always shows `>>>`) but are now in a new area that provides help.

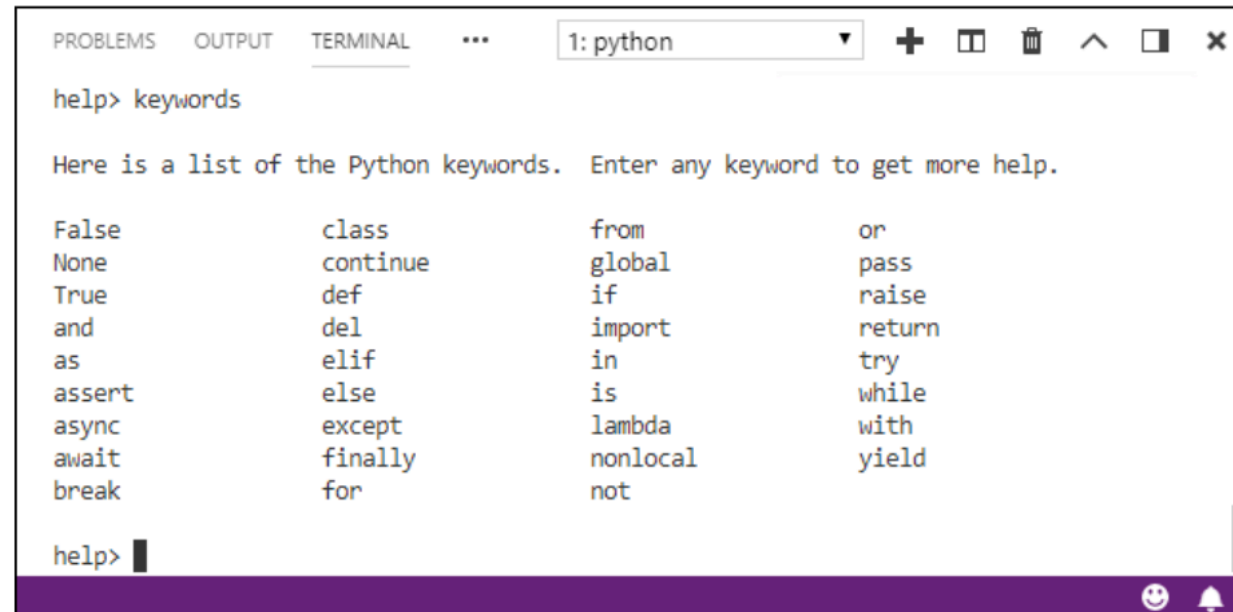
Using Python's Interactive Mode

As described on the screen, you can enter the name of any module, keyword, or topic to get help with that term. As a beginner, you might not need help with specifics right now. But it's good to know that the help is there if you need it.

For example, Python uses certain keywords, which have special meaning in the language. To get a list of those, just type the following at the `help>` prompt:

```
keywords
```

After you press Enter, you'll see a list of keywords, as shown in the following Figure



The image shows a terminal window with a dark purple title bar and a light gray background. The title bar contains the text "1: python" and several window control icons. The terminal content shows the command "help> keywords" being entered, followed by a list of Python keywords arranged in four columns. The keywords are: False, None, True, and, as, assert, async, await, break, class, continue, def, del, elif, else, except, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, raise, return, try, while, with, and yield. The prompt "help>" is followed by a cursor.

```
1: python
help> keywords

Here is a list of the Python keywords. Enter any keyword to get more help.

False      class      from       or
None       continue  global     pass
True       def        if         raise
and        del        import     return
as         elif       in         try
assert     else      is         while
async      except    lambda     with
await     finally  nonlocal   yield
break     for      not

help> |
```

FIGURE 2-4:
Keyword help.

Using Python's Interactive Mode

Exiting interactive help

¥To get out of interactive help and return to the Python prompt, type the letter `q` (for quit) or press `Ctrl+Z`. You should be back at the `>>>` prompt. At the `>>>` prompt, type `exit()` or `python`

Searching for specific help topics online

¥If you're online, you may use the web for help. If you're looking for videos, start at www.youtube.com, if not, <https://stackoverflow.com> is a good place to ask questions and search for help. And of course, there's always Google, Bing, and other search engines.

1. Creating a Python Development Workspace



Creating a Python Development Workspace |

- ¥Creating apps is easiest if you set up a VS Code development environment specifically for learning and coding Python.
- ¥VS Code uses the term workspace to define what we call a development environment. That environment is the Python interpreter you're using plus any additional extensions you gather along the way.
- ¥You can store your workspaces anywhere you like.
- ¥Follow the steps from your book page 37-39 to create your Workspace.

1. Creating a Folder for Your Python Code



Creating a Folder for Your Python Code

- ✖ Next, you create a folder to store all the Python code that you write, so it's all together in one place and easy to find when you need it. You can put this folder anywhere you like and name it whatever you like.
- ✖ Now you should associate this code folder with the VS Code workspace you just created, so that any time you work in the folder you're using the correct Python interpreter and other Python-related settings you choose over time with the files in the code folder. Here's how:
- ✖ Follow the steps from your book page 39-41 to work with your folder.



1. Typing, Editing, and Debugging Python Code



Typing, Editing, and Debugging Python Code

- ¥Most likely, you'll write most of the code in an editor. As you probably know, an editor enables you to type and edit text. Code is text. The editor in VS Code is set up for typing and editing code, so you may hear it referred to as ~~a~~ code editor
- ¥Each Python code file you create will be a plain text file with a .py filename extension. We suggest that you keep any files you create for this book in your Python folder, which you should be able to see anytime VS Code and your Python 3 workspace are open.
- ¥To create any file, follow the steps in your book page 41-42

Typing, Editing, and Debugging Python Code

| Writing Python code

- ✖ Now that you have a `.py` file open, you can use it to write some Python code. As is typical when learning a new programming language, you'll start by typing a simple Hello World program
- ✖ Follow the steps in your book page 42-43 to write your first Python code.

Typing, Editing, and Debugging Python Code

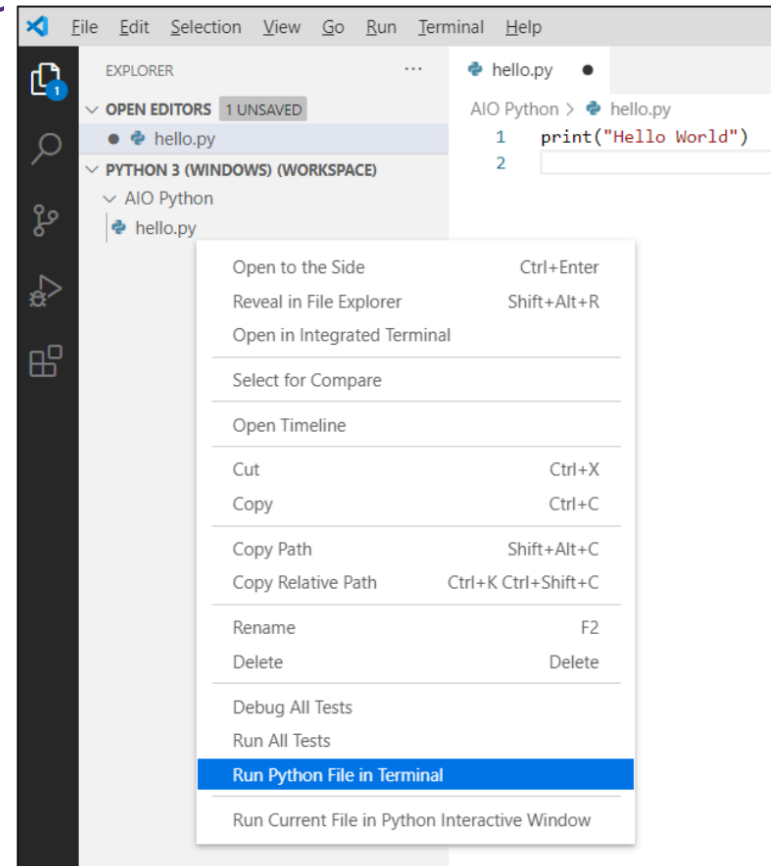
| Saving your code

- ¥Code you type in VS Code is not saved automatically. There are two ways to deal with that. One is to try to remember to save any time you make a change that's worth saving.
- ¥The second method is to use AutoSave to automatically save changes we make. To enable Auto Save, choose File! Auto Save from VS Code's menu bar. The check mark next to Auto Save means that it's turned on. To turn off AutoSave, just choose File! Auto Save again. The file is saved automatically as you make changes.

Typing, Editing, and Debugging Python Code

Running Python in VS Code

To test your Python code in VS Code, you need to run it. The easiest way to do that is to right-click the file's name (hello.py in this example) and choose Run Python File in Terminal, as shown in this Figure



Typing, Editing, and Debugging Python Code

Running Python in VS Code

¥The Terminal pane opens along the bottom of the VS Code window. You'll see a command prompt followed by a comment to run the code in the Python interpreter (`python.exe`). And below that, you'll see the output of the program: the words Hello World, in this example, and then another prompt.

Typing, Editing, and Debugging Python Code

Learning simple debugging

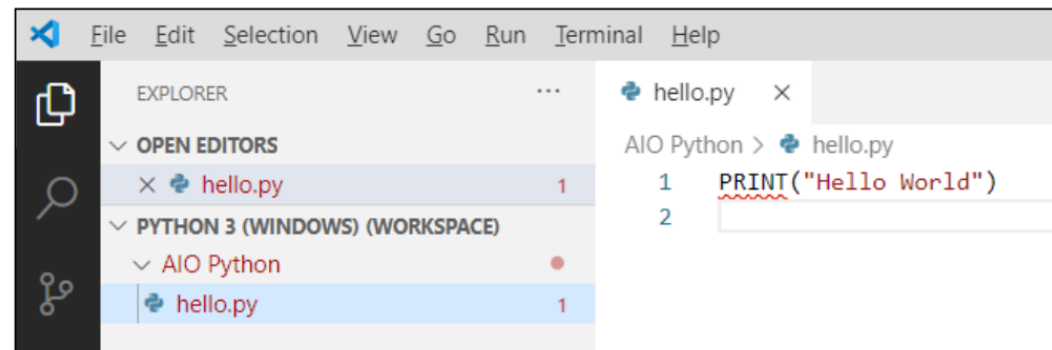
- ✖ When you're first learning to write code, you're bound to make a lot of mistakes.
- ✖ Before you attempt to run some code, you might see several screen indications of an error in your code:
 - ! The name of the folder and file that contain the error will be red in the Explorer pane.
 - ! The number of errors in the file will appear in red next to the filename in the Explorer bar.
 - ! The total number of errors will appear next to the circled X in the bottom left corner of the VS Code window.
 - ! The bad code will have a wavy red underline.

Typing, Editing, and Debugging Python Code

Learning simple debugging

In the following Figure, we typed `PRINT` in all uppercase, which is not allowed in Python. Python is case-sensitive and the correct command is `print`.

FIGURE 2-16:
PRINT is typed
incorrectly in
hello.py.



Typing, Editing, and Debugging Python Code

| Using the VS Code Python debugger

¥VS Code has a built-in debugger that helps when working with more complex programs and provides a means of testing Python programs in VS Code.

¥You won't be writing anything super complex right now. But there's no harm in getting the debugger set up and ready as of your Python development workspace. Follow the steps from your book page 46-47 to do so.



1. Writing Code in a Jupyter Notebook



Writing Code in a Jupyter Notebook

¥In week 6 materials, you learned that you can write and run Python code in a Jupyter notebook. In this section, we show you how to create, save, and open a Jupyter notebook.

| Creating a folder for Jupyter Notebook

¥A Jupyter Notebooks folder is no different from any other folder, so you can create it using whatever method you normally use in your operating system.

| Creating and saving a Jupyter notebook

¥To create a Jupyter notebook and save it in a folder, follow the steps from your book page 48.

Writing Code in a Jupyter Notebook |

| Typing and running code in a notebook

When your notebook is open, you see at least one cell. When you see Code in the drop-down menu in the toolbar below the merge bar, the active cell is for typing code. To take Jupyter Notebook for a spin, follow these steps:

1. Click in the Code cell to the right of In [] : and type the following:

```
print("Hello World")
```
2. To run the code, hold down the Alt key (Windows) or Option key (Mac)
and press Enter, or click the Run button in the toolbar above the code cell.
The output from the code appears below the cell.

Writing Code in a Jupyter Notebook |

| Adding Markdown text

- ✖ You can add text (and pictures and video) to Jupyter notebook. When typing regular text, you don't need to use any special coding. If you want to format the text or add pictures or videos however, you'll need to use Markdown tags. Markdown is a popular markup language, something like a greatly simplified HTML.
- ✖ First, make sure you're in your Jupyter Notebook app. If you have an empty cell under your Python code, choose Insert Cell Below. Then click in that new cell to add content to it. Type your text and Markdown in the cell.

Writing Code in a Jupyter Notebook |

| Adding Markdown text

- ¥To run a cell that contains Markdown, click the cell and then click Run in the toolbar. The code is rendered into text and any other content you've put in the cell.
- ¥To change code in a code cell, just click the cell and type your code normally.
- ¥With either type of cell, click Run again after making your changes. Note that only the cell that contains the cursor will run again. If you want to run all the cells in a notebook, use the double triangle icon in the toolbar.

Writing Code in a Jupyter Notebook |

| Saving and opening notebooks

¥To save a Jupyter notebook, choose File! Save and Checkpoint from the menu.

¥Any time that you want to reopen a notebook, open Anaconda and launch Jupyter Notebook. Then navigate to the file you saved and click its filename. The filename will probably have the .ipynb filename extension, which is standard for Jupyter notebooks.







Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming

Module 8:

Python Elements and Syntax



Contents

1. The Zen of Python
2. Introducing Object-Oriented Programming
3. Discovering Why Indentations Count, Big Time
4. Using Python Modules



Weekly Learning Outcomes

1. Understanding the Zen of Python
2. Introducing object-oriented programming
3. Discovering why indentations are important
4. Using Python modules



Required Reading

1. Book 1, Chapter 3. Python Elements and Syntax (John C. Shovic and Alan Simpson, Python All-in-One, 2nd edition 2021)

Videos

- ¥ Python Tutorial for Beginners | Modules
<https://youtu.be/1RuMJ53CKds>



1. The Zen of Python



The Zen of Python |

- ✖ Many programming languages focus on things that the computer does and how it does them rather than on the way humans think and work. This one simple fact makes most programming languages difficult for most people to learn.
- ✖ Python, however, is based on the philosophy that a programming language should be geared more toward how humans think, work, and communicate than what happens inside the computer.

With Python, we always try to make the code more human-readable than machine-readable.

1. Introducing Object-Oriented Programming



Introducing Object-Oriented Programming

¥There are different approaches to designing languages. Perhaps the most successful and widely used model is object-oriented programming or OOP, which is a design philosophy that tries to mimic the real world in the sense that it consists of objects with properties as well as methods (actions) that those objects perform.

¥Take a car, for example. Any one car is an object. Not all cars are exactly the same. Different cars have different properties. They all serve the same basic purpose.

¥All cars have certain methods (things they can do) in common.

Introducing Object-Oriented Programming

- ¥ You can drive them, steer them, speed them up, slow them down, control the inside temperature, and more by using controls in the car that you can manipulate with your hands.
- ¥ An object in an object-oriented programming language isn't a physical thing, like a car, because it exists only inside a computer.
- ¥ An object is strictly a software thing. In Python, you can have a class (which you can think of as an object creator, such as a car factory) that can produce many kinds of objects (cars) for varying purposes (sporty, off-road, sedan).

Introducing Object-Oriented Programming

- ¥All these objects can be manipulated through the controls they all have in common, much as all cars are manipulated by controls such as the steering wheel, brakes, accelerator, and gearshift.
- ¥Python is very much an object-oriented language. The core language consists of controls (in the form of words) that allow you to control all different kinds of objects Ñ in your own and other peoplesÕ programs.
- ¥However, you need to learn the core language first so that when youÕre ready to start using other peoplesÕ objects, you know how to do so.

1. Discovering Why Indentations Count, Big Time



Discovering Why Indentations Count, Big Time

- ✖ In terms of the basic style of writing code, the one feature that really makes Python different from other languages is that it uses indentations rather than parentheses and curly braces as such to indicate blocks, or chunks, of code.
- ✖ For example, here's some JavaScript code. If you're familiar with the Magic 8 Ball toy, you may have a sense of what this program is doing. But that's not what's important. Just note those parentheses, curly braces, and semicolons:

Discovering Why Indentations Count, Big Time

```
document.addEventListener("DOMContentLoaded", function () {var question =  
    prompt("Ask magic 8 ball a question");var answer = Math.floor(Math.random() *  
    8) + 1; if (answer == 1) {alert("It is certain");} else if (answer == 2)  
    {alert("Outlook good");} else if (answer == 3) {alert("You may rely on it");}  
    else if (answer == 4) {alert("Ask again later");} else if (answer == 5)  
    {alert("Concentrate and ask again");} else if (answer == 6) {alert ("Reply  
    hazy, try again");} else if (answer == 7) {alert("My reply is no");} else if  
    (answer == 8) {alert("My sources say no")} else {alert ("That's not a  
    question");}alert("The end");})
```

¥The code is a mess and not fun to read. We can make reading it a little easier by breaking it into multiple lines and indenting some of those lines. (Note that doing so isn't required in JavaScript.) Following is the reformatted code:

Discovering Why Indentations Count, Big Time

```
document.addEventListener("DOMContentLoaded", function () {  
    var question = prompt("Ask magic 8 ball a question");  
    var answer = Math.floor(Math.random() * 8) + 1;  
    if (answer == 1) {  
        alert("It is certain");  
    } else if (answer == 2) {  
        alert("Outlook good");  
    } else if (answer == 3) {  
        alert("You may rely on it");  
    } else if (answer == 4) {  
        alert("Ask again later");  
    } else if (answer == 5) {  
        alert("Concentrate and ask again");  
    } else if (answer == 6) {  
        alert("Reply hazy, try again");  
    } else if (answer == 7) {  
        alert("My reply is no");  
    } else if (answer == 8) {  
        alert("My sources say no")  
    } else {  
        alert("That's not a question");  
    }  
    alert("The end");  
})
```

Discovering Why Indentations Count, Big Time

- ¥In JavaScript, the parentheses and curly braces are required because they identify where chunks of code begin and end. The indentations for readability are optional.
- ¥The rules are opposite in Python because it doesn't use curly braces or any other special characters to mark the beginning and end of a block of code.
- ¥The indentations themselves mark those. So those indentations aren't optional Ñ they are required and have a considerable effect on how the code runs.

Discovering Why Indentations Count, Big Time

¥As a result, when you read the code (as a human, not as a computer), it's relatively easy to see what's going on, and you're not distracted by a ton of extra quotation marks. Here is that JavaScript code written in Python:

```
import random
question = input("Ask magic 8 ball a question")
answer = random.randint(1, 8)
if answer == 1:
    print("It is certain")
elif answer == 2:
    print("Outlook good")
elif answer == 3:
```

Discovering Why Indentations Count, Big Time

```
    print("You may rely on it")
elif answer == 4:
    print("Ask again later")
elif answer == 5:
    print("Concentrate and ask again")
elif answer == 6:
    print("Reply hazy, try again")
elif answer == 7:
    print("My reply is no")
elif answer == 8:
    print("My sources say no")
else:
    print("That's not a question")
print("The end")
```

¥You may have noticed at the top of the Python code the line that starts with import. Lines that start with import are common in Python, and you'll see why in the next section.

1. Using Python Modules



Using Python Modules

- ¥One of the secrets to Python's success is that it's comprised a simple, clean, core language. That's the part you need to learn first.
- ¥In addition to that core language, many, many modules are available that you can grab for free and access from your own code.
- ¥These modules are also written in the core language, but you don't need to see that or even know it because you can access all the power of the modules from the basic core language.

Using Python Modules

- ¥Most modules are for some a specific application such as science or artificial intelligence or working with dates and time or whatever.
- ¥The beauty of using modules is that other people spent a lot of time creating, testing, and finetuning that module so you don't have to.
- ¥You simply import the module into your own Python file, and use the module's capabilities as instructed in the module's documentation.

Using Python Modules

¥The preceding sample Magic 8 Ball program starts with this line

```
import random
```

¥The core Python language has nothing built into it to generate random number.

¥Although we could figure out a way to make a random number generator, we don't need to because someone has figured out how to do it and has made the code freely available.

¥Starting your program with `import random` tells the program that you want to use the capabilities of the random number module to generate a random number.

Using Python Modules

Then, later in the program, you generate a random number between 1 and 8 with this line of code:

```
answer = random.randint(1, 8)
```

Hundreds of free modules for Python are available Ñ you just need to know which ones to import into your program.

Using Python Modules

- ¥You may notice that some version numbers are colored and are preceded with an arrow, which indicates that a more recent version may be available for you to download.
- ¥As with programming languages, modules evolve over time as their authors improve them and add new capabilities. You're not required to have the latest version, though. If the version you have is working, you might want to stick with it.

Understanding the syntax for importing modules

As mentioned, in your own Python code, you must import a module before you can access its capabilities. The syntax for doing so is

```
import modulename [as alias]
```

Code written in a generic format like that, with some parts in *italic*, some in square brackets, is sometimes called a syntax c because it's not showing you, literally, what to type. Rather, it's showing the syntax (format) of the code.

Understanding the syntax for importing modules

Here is how information is presented in such a syntax chart:

- ! The code is case-sensitive, meaning you must `import` and as using all lowercase letters, as shown.
- ! Anything in italics is a placeholder for information you should supply in your own code.
- ! Anything in square brackets is optional
- ! You never type the square brackets in your code because they are part of the Python language.

Understanding the syntax for importing modules

Here is how information is presented in such a syntax chart:

- ! The code is case-sensitive, meaning you must `import` and as using all lowercase letters, as shown.
- ! Anything in italics is a placeholder for information you should supply in your own code.
- ! Anything in square brackets is optional
- ! You never type the square brackets in your code because they are part of the Python language.

Note: Always put import statements first, so their capabilities are available to the rest of the code.

Using an alias with modules

As you just saw with the import command's syntax, you can assign an alias, or nickname, to any module you import just by following the module name with a space, the word `as`, and a name of your own choosing.

For example, instead of typing `import random` to import that module, you could import it and give it a nickname such as `rnd`, which is shorter:

```
import random as rnd
```

Using Python Modules

Using an alias with modules

Then, in subsequent code, you wouldn't use the full name, `random`, to refer to the module. Instead, you'd use the short name, `rnd`:

```
answer = rnd.randint(1, 8)
```

Using an alternative short name may not seem like a big deal in this short example. But some modules have lengthy names.





Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming

Module 9:

Building Your First Python Application



Contents

1. Opening the Python App File
2. Typing and Using Python Comments
3. Understanding Python Data Types
4. Working with Python Operators
5. Creating and Using Variables
6. Understanding What Syntax Is and Why It Matters
7. Putting Code Together



Weekly Learning Outcomes

1. Opening the Python file
2. Using Python comments
3. Understanding data types in Python
4. Doing work with Python operators
5. Creating variables
6. Understanding syntax
7. Organizing code



Required Reading

1. Book 1 Chapter 4. Building Your First Python Application (John C.Shovicand Alan Simpson, Python All-in-Oneed^{ed} edition 2021)

Videos

- ¥ Python Tutorial for Beginners | Data Types in Python
<https://youtu.be/gCCVsvgR2KU>
- ¥ Python Tutorial for Beginners | Operators in Python
<https://youtu.be/v5MR5JnKcZI>



Introduction



Introduction |

- ¥Like any language, you need to understand the individual words so that you can start building sentences and, finally, the blocks of code that will enable your app to work.
- ¥First, we walk you through creating an app file in which you will create your code. Then you learn the various data types, operators, and variables, which are the words of the Python language, and then Python syntax.
- ¥Along the way, you see how to save your app, catch mistakes, and comment your code so that you and others can understand how you built it and why.

1. Opening the Python App File



Opening the Python App File

¥We assume that you've already set up your learning and development environment, as described in previous chapters. And you know how to open the main tools, Anaconda Navigator and VS Code .

¥Follow these steps:

- ! Open Anaconda Navigator and launch VS Code from there.
- ! If your Python 3 workspace doesn't open automatically, choose "File" > "Open Workspace" from the VS Code menu and open the Python 3 workspace you created before.

Opening the Python App File

- ! Click the `hello.py` file you created in previous Chapter.
- ! Select all the text on the first line and delete it, so you can start from scratch.

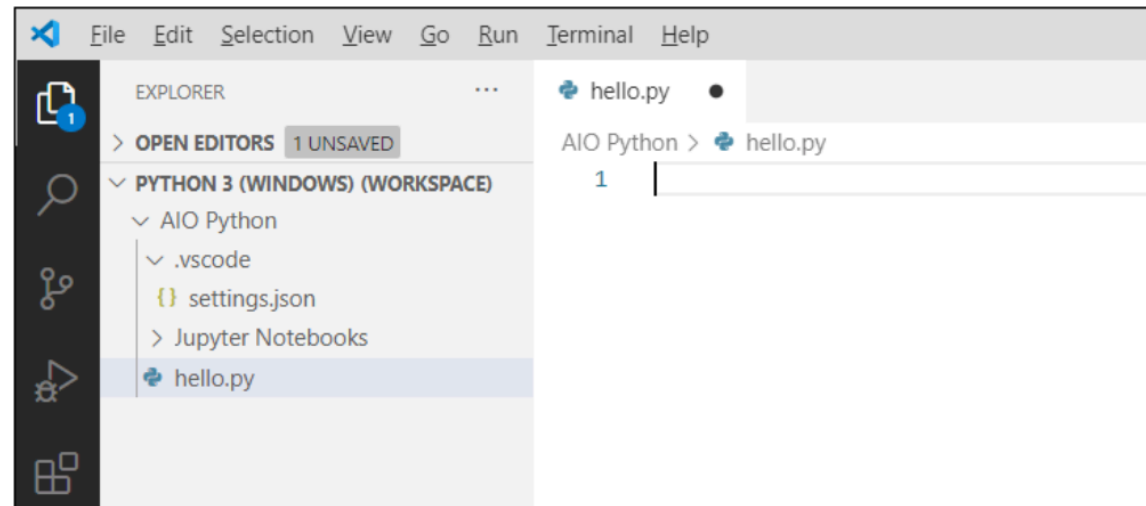


FIGURE 4-1:
The `hello.py` file, open for editing in VS Code.

1. Typing and Using Python Comments



Typing and Using Python Comments

- ¥A comments a text in the program that does nothing.
- ¥When working in teams, professionals often use comments to explain to team members what their code is doing.
- ¥Developers will also put comments in their code as notes to themselves, so that if they review the code in the future, they can refer to their own notes for reminders on why they did something in the code.

Typing and Using Python Comments

¥To be identified as a comment, you must do one of the following:

- ! Start the text with a pound sign (#)

- ! Enclose the text in triple quotation marks

¥If the comment is short (one line), the # sign is sufficient. After youÕll see the # sign followed by a space, but the space is optional. The comment text will be green if youÕre using the default color theme.

```
# This is a Python comment
```

Typing and Using Python Comments

¥You can type longer comments in Python by enclosing them in triple quotation marks.

```
"""This is a multiline comment in Python  
This type of comment is sometimes called a docstring.  
A docstring starts with three double-quotation marks, and also ends with three  
double quotation marks. """
```

¥You can have an unlimited number of comments in your code.

1. Understanding Python Data Types



Understanding Python Data Types

¥Numbers are amounts, such as 10 or 123.45.

¥Text consists of letters and words.

¥For computers, the big difference is that they can do arithmetic (add, subtract, multiple, divide) with numbers, but not with letters and words.

Understanding Python Data Types

| Numbers

- ¥Numbers in Python must start with a number digit, (0-9); a dot (period), which is a decimal point; or a hyphen (-) used as a negative sign for negative numbers.
- ¥A number can contain only one decimal point. It should not contain letters, spaces, dollar signs, or anything else that isn't part of a normal number.
- ¥The following Table shows example of good and bad Python numbers.

TABLE 4-1**Examples of Good and Bad Python Numbers**

Number	Good or Bad?	Reason
1	Good	A whole number (integer)
1.1	Good	A number with a decimal point
1234567.89	Good	A large number with a decimal point and no commas
-2	Good	A negative number, as indicated by the starting hyphen
.99	Good	A number that starts with a decimal point because it's less than 1
\$1.99	Bad	Contains a \$
12,345.67	Bad	Contains a comma
1101 3232	Bad	Contains a space
91740-3384	Bad	Contains a hyphen
123-45-6789	Bad	Contains two hyphens
123 Oak Tree Lane	Bad	Contains spaces and words
(267)555-1234	Bad	Contain parentheses and hyphens
127.0.0.1	Bad	Only one decimal point is allowed

Understanding Python Data Types

| Numbers

- ¥Integers: An integer is any whole number, positive or negative. There is no limit to its size. Numbers such as 0, -1, and 9999999999999999 are all perfectly valid integers (that doesn't contain a decimal point).
- ¥Floats: A floating-point number, is any valid number that contains a decimal point. There is no size limit: 1.1 and -1.1 and 123456.789012345 are all perfectly valid floats.
- ¥Complex numbers Python also supports complex numbers. These bizarre little charmers always end with the letter *j*, which is the imaginary part of the number.

Understanding Python Data Types

| Words (strings)

- ¥Names, addresses, and all other kinds of text you see every day would be a string in Python.
- ¥To a computer, if a piece of information is not something on which it can do arithmetic, it's just a string of characters.
- ¥Unlike numbers, a string must always be enclosed in quotation marks. You can use either double (") or single (') quotation marks. All the following are valid strings:

Understanding Python Data Types

| Words (strings)

```
"Hi there, I am a string"
```

```
'Hello world'
```

```
"123 Oak Tree Lane"
```

```
"(267)555-1234"
```

```
"18901-3384"
```

¥Note that it's fine to use numeric characters (0-9) as well as hyphens and dots (periods) in strings. Each is still a string because it's enclosed in quotation marks.

Understanding Python Data Types

| Words (strings)

¥A word of caution. If a string contains an apostrophe (single quote), the entire string should be enclosed in double quotation marks like this:

```
"Mary's dog said Woof"
```

¥Similarly, if the string contains double quotation marks, enclose the entire thing in single quotation marks to avoid confusion. For example:

```
'The dog of Mary said "Woof".'
```

Understanding Python Data Types

| Booleans

¥A third data type in Python isn't exactly a number or a string. It's called **Boolean**, and it can be one of two values: either **True** or **False**

¥In Python code, people store **True** and **False** values in variables (placeholders in code that we discuss later in this chapter) using a format like this:

```
x = True
```

¥Or perhaps this:

```
x = False
```

Understanding Python Data Types

| Booleans

¥You know `True` and `False` are Boolean here because they are not enclosed in quotation marks (as a string would be) and are not numbers.

¥Also, the initial cap is required. In other words, the Boolean values `True` and `False` must be written as shown.



1. Working with Python Operators



Working with Python Operators

- Python offers many different operators for working with and comparing types of information.
- Here we summarize them all for future reference, without going into great detail.
- Whether you use an operator in your own work depends on the types of apps you develop.

Working with Python Operators

Arithmetic operators

Arithmetic operators, as the name implies, are for doing arithmetic; addition, subtraction, multiplication, division, and more.

This Table lists Python's arithmetic operators.

Python's Arithmetic Operators

Operator	Description	Example
+	Addition	1 + 1 = 2
-	Subtraction	10 - 1 = 9
*	Multiplication	3 * 5 = 15
/	Division	10 / 5 = 2
%	Modulus (remainder after division)	11 % 5 = 1
**	Exponent	3**2 = 9
//	Floor division	11 // 5 = 2

Working with Python Operators

Comparison operators

¥The comparison operators Python offers to help you write code that makes decisions are listed in Table below.

Python Comparison Operators

Operator	Meaning
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
!=	Not equal to
is	Object identity
is not	Negated object identity

Working with Python Operators

Boolean operators

The Boolean operators work with Boolean values (True or False) and are used to determine if one or more things are True or False. The Table below summarizes the Boolean operators.

Python Boolean Operators

Operator	Code Example	What It Determines
or	x or y	Either x or y is True
and	x and y	Both x and y are True
not	not x	x is not True

1. Creating and Using Variables



Creating and Using Variables

Creating valid variable names

¥A variable is simply a placeholder for information that may vary (change). In your code, a variable is represented by a **variable**.

¥In Python, variable must conform to the following rules to be recognized as variable names:

- ! The variable name must start with a letter or an underscore
- ! After the first character, you can use letters, numbers, or underscores.

Creating and Using Variables

Creating valid variable names

- ! Variable names are case sensitive, so after you make up a name, any reference to that variable must use the same uppercase and lowercase letters.
- ! Variable names cannot be enclosed in, or contain, single or double quotation marks.
- ! PEP 8 style conventions recommend that you use only lowercase letters in variable names and use an underscore to separate multiple words.

Creating and Using Variables

Creating variables in code

¥To create a variable, you use the following:

```
variablename = value
```

¥where `variablename` is the name you make up. You can use `x` or `y`, as people often do in math. But in larger programs, it's a good idea to give your variables more meaningful names, such as `quantity` or `unit_price` or `sales_tax`, so that you can remember what you're storing in the variable.

¥The `value` is whatever you want to store in the variable. It can be a number, a string, or a Boolean `True` or `False` value.

Creating and Using Variables

Creating variables in code

¥The = sign is the assignment operator and is so named because it assigns the value (on the right) to the variable (on the left). For example, in the following:

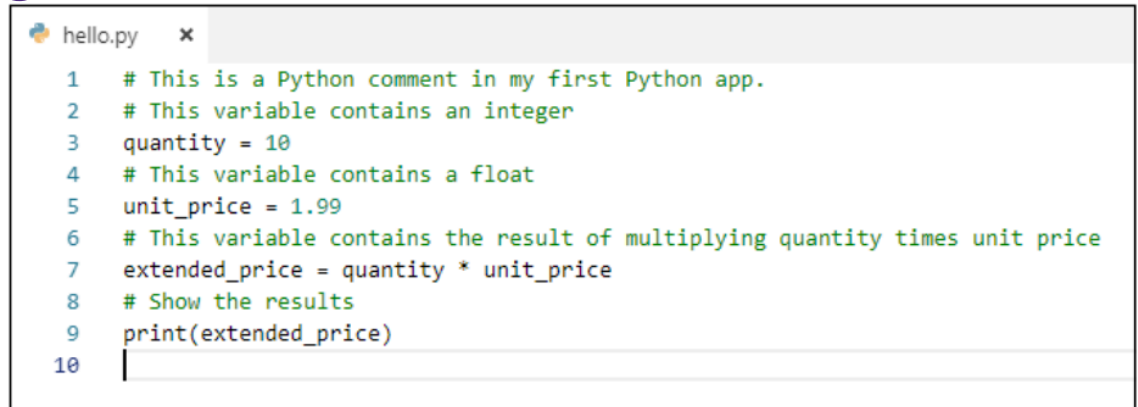
```
x = 10
```

¥We are storing the number 10 in a variable named x. In other words, we are assigning the value 10 to the variable.

Creating and Using Variables

Manipulating variables

FIGURE 4-4:
Your first Python
app typed into VS
Code.



```
hello.py x
1  # This is a Python comment in my first Python app.
2  # This variable contains an integer
3  quantity = 10
4  # This variable contains a float
5  unit_price = 1.99
6  # This variable contains the result of multiplying quantity times unit price
7  extended_price = quantity * unit_price
8  # Show the results
9  print(extended_price)
10 |
```

- ¥Your Python app creates some variables, stores some values in them, and calculates a new value `extended_price` based on the contents of the `quantity` and `unit_price` variables.
- ¥The last line displays the contents of the `extended_price` variable on the screen.

Creating and Using Variables

| Saving your work

¥ Typing code is like typing other documents on a computer. If you don't save your work, you may not have it the next time you sit down at your computer and go looking for it.

| Running your Python app in VS Code

¥ Now you can run the app and see if it works. An easy way to do that is to right-click the `hello.py` filename in the Explorer bar and choose Run Python File in Terminal.

¥ If your code is typed correctly, you should see the result, 19.9, in the Terminal window,

1. Understanding What Syntax Is and Why It Matters



Understanding What Syntax Is and Why It Matters

- ¥ Syntax is important in human languages because order contributes much to the meaning.
- ¥ Proper syntax in programming languages is every bit as important as it is in human languages.
- ¥ The `SyntaxError` part tells you that Python doesn't know what to do with that line of code because you didn't follow the proper syntax.

Understanding What Syntax Is and Why It Matters

Now let's talk about individual lines of code. In Python, a line of code ends with a line break or a semicolon. For example, this is three lines of Python code:

```
first_name= "Alan"  
last_name= "Simpson"  
print(first_name, last_name)
```

It would also be acceptable to use a semicolon instead of a line break:

```
first_name= "Alan"; last_name= "Simpson"  
print(first_name, last_name)
```

Understanding What Syntax Is and Why It Matters

Or, if you prefer:

```
first_name = "Alan"; last_name = "Simpson"; print(first_name, last_name)
```

The code runs the same whether you end each line with a break or a semicolon.



1. Putting Code Together



Putting Code Together

- ¥ The exercises you've just completed explain how to type, save, run, and change an app, save it again, and run it again.
- ¥ Those tasks define what you'll be doing with any kind of software development in any language, so you should practice them until they become second nature.



Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming
Module 10:
Working with Numbers, Text, and Dates
-PART 1-



Contents

Part 1

1. Calculating Numbers with Functions
2. Still More Math Functions
3. Formatting Numbers
4. Grappling with Weirder Numbers



Contents

Part 2

1. Manipulating Strings
2. Uncovering Dates and Times
3. Accounting for Time Zones
4. Working with Time Zones



Weekly Learning Outcomes

1. Mastering whole numbers
2. Juggling numbers with decimal points



Required Reading

1. Book 2, Chapter 1. Working with Numbers, Text, and Dates (John CShovicand Alan Simpson, Python All-in-One 2nd edition 2021)

Videos

- ¥ Built-in Python Functions That Improves Your Coding | Python Tutorial
<https://youtu.be/7p1Gqwf-56s>
- ¥ Python: Dates, Times & Timestamps Part-1 | datetime, time libraries
<https://youtu.be/hUkDhkrUNys>



Part 1



Introduction



Introduction |

- ¥In the computer world, numbers are numbers you can add, subtract, multiply, and divide.
- ¥Python also differentiates between whole numbers (integers) and numbers that contain a decimal point (floats).
- ¥Words (textual information such as names and addresses) are stored as strings, which is short for "a string of characters."
- ¥In addition to numbers and strings, there are Boolean values, which can be either True or False

Introduction |

¥In real life, we also have to deal with dates and times, which are yet another type of information.

¥Python doesn't have a built-in data type for dates and times, but thankfully, a free module you can import any time works with such information.

¥This chapter is all about taking full advantage of the various Python data types.



1. Calculating Numbers with Functions



Calculating Numbers with Functions

Python functions generally have the syntax:

```
variablename = functionname(param[,param])
```

Because most functions return some value, you typically start by defining a variable to store what the function returns.

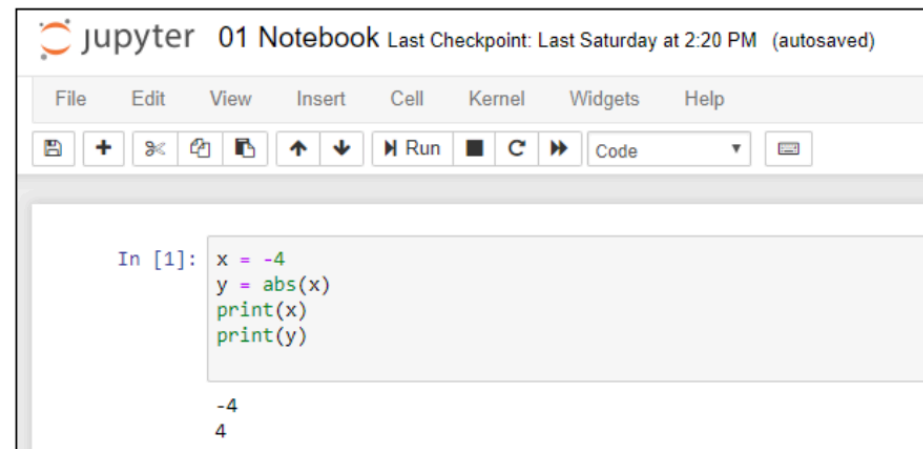
Follow that with the = sign and the function name, followed by a pair of parentheses.

Inside the parentheses you may pass one or more values (called parameters) to the function.

For example, the `abs()` function accepts one number and returns the absolute value of that number.

Calculating Numbers with Functions

¥For example, the `abs()` function accepts one number and returns the absolute value of that number.

A screenshot of a Jupyter Notebook interface. The title bar says "jupyter 01 Notebook" and "Last Checkpoint: Last Saturday at 2:20 PM (autosaved)". The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu is a toolbar with icons for saving, adding cells, undo, redo, and running code. The main area shows a code cell with the following Python code:

```
In [1]: x = -4
        y = abs(x)
        print(x)
        print(y)
```

The output of the code cell is displayed below the code:

```
-4
4
```

FIGURE 1-1:
Trying out the
`abs()` function.

¥Some functions accept two or more values. For example, the `round()` function takes one number as its first argument. The second argument is the number of decimal places.

TABLE 1-1 **Some Built-In Python Functions for Numbers**

Built-In Function	Purpose
<code>abs(x)</code>	Returns the absolute value of number <i>x</i> (converts negative numbers to positive).
<code>bin(x)</code>	Returns a string representing the value of <i>x</i> converted to binary.
<code>float(x)</code>	Converts a string or number <i>x</i> to the float data type.
<code>format(x, y)</code>	Returns <i>x</i> formatted according to the a pattern specified in <i>y</i> . This older syntax has been replaced with f-strings in current Python versions.
<code>hex(x)</code>	Returns a string containing <i>x</i> converted to hexadecimal, prefixed with 0x.
<code>int(x)</code>	Converts <i>x</i> to the integer data type by truncating (not rounding) the decimal portion and any digits after it.
<code>max(x, y, z, ...)</code>	Takes any number of numeric arguments and returns whichever is the largest.
<code>min(x, y, z, ...)</code>	Takes any number of numeric arguments and returns whichever is the smallest.
<code>oct(x)</code>	Converts <i>x</i> to an octal number, prefixed with 0o to indicate octal.
<code>round(x, y)</code>	Rounds the number <i>x</i> to <i>y</i> number of decimal places.
<code>str(x)</code>	Converts the number <i>x</i> to the string data type.
<code>type(x)</code>	Returns a string indicating the data type of <i>x</i> .

Calculating Numbers with Functions

¥Examples of proper Python syntax for using the built-in math functions.

```
: pi=3.14159265358979
x=128
y=-345.67890987
z=-999.9999
print(abs(z))
print(int(z))
print(int(abs(z)))
print(round(pi,4))
print(bin(x))
print(hex(x))
print(oct(x))
print(max(pi,x,y,z))
print(min(pi,x,y,z))
print(type(pi))
print(type(x))
print(type(str(y)))

999.9999
-999
999
3.1416
0b10000000
0x80
0o200
128
-999.9999
<class 'float'>
<class 'int'>
<class 'str'>
```



1. Still More Math Functions



Still More Math Functions

- ¥ In addition to the built-in functions you've learned about so far, still others you can import from the math module.
- ¥ If you need them in an app, put `import math` near the top of the .py file or Jupyter cell to make those functions available to the rest of the code
- ¥ One of the functions in the math module is the `sqrt()` function, which gets the square root of a number.
- ¥ Because it's part of the math module, you can't use it without importing the module first.

Still More Math Functions

- ¥To use a function from a module, you have to import the module and precede the function name with the module name and a dot.
- ¥So let's say you have some value, x , and you want the square root. You have to import the math module and use `math.sqrt(x)` to get the correct answer as shown in the following Figure.
- ¥Entering that command shows 9.0 as the result, which is indeed the square root of 81.

Still More Math Functions

FIGURE 1-4:
Using the `sqrt()`
function from the
`math` module.

```
In [11]: import math
          z = 81
          print(math.sqrt(81))

          9.0
```

¥The `math` module offers a lot of trigonometric and hyperbolic functions, powers and logarithms, angular conversions, constants such as π and e .

Still More Math Functions

TABLE 1-2 **Some Functions from the Python Math Module**

Built-In Function	Purpose
<code>math.acos(x)</code>	Returns the arccosine of <code>x</code> in radians
<code>math.atan(x)</code>	Returns the arctangent of <code>x</code> , in radians
<code>math.atan2(y, x)</code>	Converts rectangular coordinates (<code>x</code> , <code>y</code>) to polar coordinates (<code>r</code> , <code>theta</code>)
<code>math.ceil(x)</code>	Returns the ceiling of <code>x</code> , the smallest integer greater than or equal to <code>x</code>
<code>math.cos(x)</code>	Returns the cosine of <code>x</code> radians
<code>math.degrees(x)</code>	Converts angle <code>x</code> from radians to degrees
<code>math.e</code>	Returns the mathematical constant <i>e</i> (2.718281 . . .)
<code>math.exp(x)</code>	Returns <i>e</i> raised to the power <code>x</code> , where <i>e</i> is the base of natural logarithms

(continued)

Still More Math Functions

TABLE 1-2 *(continued)*

Built-In Function	Purpose
<code>math.factorial(x)</code>	Returns the factorial of <code>x</code>
<code>math.floor()</code>	Returns the floor of <code>x</code> , the largest integer less than or equal to <code>x</code>
<code>math.isnan(x)</code>	Returns <code>True</code> if <code>x</code> is not a number; otherwise returns <code>False</code>
<code>math.log(x, y)</code>	Returns the logarithm of <code>x</code> to base <code>y</code>
<code>math.log2(x)</code>	Returns the base-2 logarithm of <code>x</code>
<code>math.pi</code>	Returns the mathematical constant pi (3.141592 . . .)
<code>math.pow(x, y)</code>	Returns <code>x</code> raised to the power <code>y</code>
<code>math.radians(x)</code>	Converts angle <code>x</code> from degrees to radians
<code>math.sin(x)</code>	Returns the sine of <code>x</code> , in radians
<code>math.sqrt(x)</code>	Returns the square root of <code>x</code>
<code>math.tan(x)</code>	Returns the tangent of <code>x</code> radians
<code>math.tau()</code>	Returns the mathematical constant tau (6.283185 . . .)

Still More Math Functions

FIGURE 1-5:
More playing
around with
built-in math
functions at the
Python prompt.

```
In [22]: import math
         pi = math.pi
         e = math.e
         tau = math.tau
         x = 81
         y = 7
         z = -23234.5454
         print(pi)
         print(e)
         print(tau)
         print(math.sqrt(x))
         print(math.factorial(y))
         print(math.floor(z))
         print(math.degrees(y))
         print(math.radians(45))

3.141592653589793
2.718281828459045
6.283185307179586
9.0
5040
-23235
401.07045659157626
0.7853981633974483
```



1. Formatting Numbers



Formatting Numbers

Formatting with f-strings

Format strings, or f-strings, are the easiest way to format data in Python. All you need is a lowercase `f` or uppercase `F` followed immediately by some text or expressions enclosed in quotation marks. Here is an example:

FIGURE 1-6:
A super simple
f-string for
formatting.

```
In [24]: username = "Alan"
          print(f"Hello {username}")
          Hello Alan
```

Formatting Numbers

| Formatting with f-strings

- ¥ The `f` before the first quotation mark tells Python that what follows is a format string.
- ¥ Inside the quotation marks, the text, called the *literal part*, is displayed literally (exactly as typed in the string). Anything in curly braces is the *expression part* of the f-string, a placeholder for what will appear when the code executes.
- ¥ Inside the curly braces, you can have an *expression* (a formula to perform some calculation, a variable name, or a combination of the two).

Formatting Numbers

Showing dollar amounts

¥To get a comma to appear in the dollar amount and the pennies as two digits, you can use a format string inside the curly braces of an expression in an f-string. The format string starts with a colon and needs to be placed inside the closing curly brace, right up against the variable name or the value shown.

¥To show commas in thousands places, use a comma in your format string right after the colon. Example:

```
print(f"Subtotal ${quantity * unit_price:,.2f}")
```

¥Here's what the code displays when executed:

```
Subtotal: $1,499.70
```

Formatting Numbers

Showing dollar amounts

¥To get a comma to appear in the dollar amount and the pennies as two digits, you can use a format string inside the curly braces of an expression in an f-string. The format string starts with a colon and needs to be placed inside the closing curly brace, right up against the variable name or the value shown.

¥To show commas in thousands places, use a comma in your format string right after the colon. The `%.2f` means "two decimal places, fixed" (never any more or less than two decimal places).

¥Example:

```
print(f"Subtotal ${quantity * unit_price:,.2f}")
```

¥Here's what the code displays when executed:

```
Subtotal: $1,499.70
```

Formatting Numbers

Formatting percent numbers

When displaying the sales tax rate for people to read, you'll probably want to use the more familiar 6.5% format rather than .065. You can use the same idea as with fixed numbers (.2f). However, you replace the f for fixed numbers with %, like this:

FIGURE 1-7:
Formatting a
percentage
number.

```
In [36]: sales_tax_rate = 0.065  
         print(f"Sales Tax Rate {sales_tax_rate:.2%}")  
  
Sales Tax Rate 6.50%
```

Formatting Numbers

| Making multiline format strings

¥If you want to have multiline output, you can add line breaks to your format strings in a few ways:

¥Use `/n`: You can use a single-line format string with `\n` in any place you want a line break. Just make sure you put the `\n` in the literal portion of the format string, not inside curly braces. For example:

<code>user1 = "Alberto"</code>	Alberto
<code>user2 = "Babs"</code>	Babs
<code>user3 = "Carlos"</code>	Carlos
<code>output=f"{user1} \n{user2} \n{user3}"</code>	
<code>print(output)</code>	

Formatting Numbers

Making multiline format strings

Use triple quotation marks (single or double): If you use triple quotation marks around your format string, you don't need to use `\n`. You can just break the line in the format string wherever you want it to break in the output. The format string is in triple quotation marks and contains multiple line breaks. The output from running the code has line breaks in the same places.

```
user1 = "Alberto"
user2 = "Babs"
user3 = "Carlos"
output=f"{user1} \n{user2} \n{user3}"
print(output)
```

this code displays:

```
Alberto
Babs
Carlos
```

Formatting Numbers

Formatting width and alignment

- ¥You can also control the width of your output (and the alignment of content within that width) by following the colon in your f-string with `<` (for left aligned), `^` (for centered), or `>` (for right aligned).
- ¥Put any of these characters right after the colon in your format string.
- ¥For example, in the next Figure, each f-string includes `>9,` which causes each displayed number to be right aligned and 9 characters wide.

Formatting Numbers

| Formatting width and alignment

- ¥The output, which you can see at the bottom of the figure, makes all the numbers align to the right, with their dollar sign neatly aligned to the left.
- ¥The spaces to the right of each dollar sign make sure each number is exactly 9 characters wide.
- ¥You may wonder why the dollar signs are lined up the way they are. Why aren't they aligned right next to their numbers? The dollar signs are part of the literal string, outside the curly braces, so they aren't affected by the `>9` inside the curly braces.

FIGURE 1-10:
All dollar
amounts are right
aligned within
a width of
9 characters (>9).

```
unit_price = 49.95
quantity = 32
sales_tax_rate = 0.065
subtotal = quantity * unit_price
sales_tax = sales_tax_rate * subtotal
total = subtotal + sales_tax
output=f"""
Subtotal:  ${subtotal:>9,.2f}
Sales Tax: ${sales_tax:>9,.2f}
Total:     ${total:>9,.2f}
"""
print(output)
```

```
Subtotal:  $ 1,598.40
Sales Tax: $   103.90
Total:     $ 1,702.30
```

1. Grappling with Weirder Numbers



Grappling with Weirder Numbers

| Binary, octal, and hexadecimal numbers

¶ If your work requires dealing with base 2, base 8, or base 16 numbers, you're in luck because Python has symbols for writing these as well as functions for converting among them

TABLE 1-3 Python for Base 2, 8, and 16 Numbers

System	Also Called	Digits Used	Symbol	Function
Base 2	Binary	0,1	0b	bin()
Base 8	Octal	0,1,2,3,4,5,6,7	0o	oct()
Base 16	Hexadecimal or hex	0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F	0x	hex()

Grapppling with Weirder Numbers

Binary, octal, and hexadecimal numbers

¥To convert from binary, octal, or hex to decimal, you don't need to use a function. Just use `print()` with the number you want to convert inside the parentheses.

```
x=255
# Convert decimal to other number systems
print(bin(x))
print(oct(x))
print(hex(x))

# Show number in decimal number system (no conversion required)
print(0b11111111)
print(0o377)
print(0xff)

0b11111111
0o377
0xff
255
255
255
```

FIGURE 1-12:
Messing about
with binary, octal,
and hex.



Thank You





الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432

College of Computing and Informatics

Introduction to Data Science Programming



Introduction to Data Science Programming
Module 11:
Working with Numbers, Text, and Dates
-PART 2-



Contents

Part 1

1. Calculating Numbers with Functions
2. Still More Math Functions
3. Formatting Numbers
4. Grappling with Weirder Numbers



Contents

Part 2

1. Manipulating Strings
2. Uncovering Dates and Times
3. Accounting for Time Zones
4. Working with Time Zones



Weekly Learning Outcomes

1. Simplifying strings
2. Working with dates and times



Required Reading

1. Book 2, Chapter 1. Working with Numbers, Text, and Dates (John CShovicand Alan Simpson, Python All-in-One 2nd edition 2021)

Videos

- ¥ Built-in Python Functions That Improves Your Coding | Python Tutorial
<https://youtu.be/7p1Gqwf-56s>
- ¥ Python: Dates, Times & Timestamps Part-1 | datetime, time libraries
<https://youtu.be/hUkDhkrUNys>



Part 2



1. Manipulating Strings



Manipulating Strings

Concatenating strings

- You can join strings by using a + sign. The process of doing so is called string concatenation.
- When you join strings, the computer doesn't automatically put in spaces where you'd expect them.
- Because Python won't automatically put in spaces where you think they should go, you have to put them in yourself.
- The easiest way to represent a single space is by using a pair of quotation marks with one space between them

Manipulating Strings

Concatenating strings

```
first_name= "Alan"  
middle_init= "C"  
last_name= "Simpson"  
full_name= first_name+ " " + middle_init+ ". " +  
last_name  
print(full_name)
```

¥The output of this code, which is the contents of that `full_name` variable

Alan C. Simpson

Manipulating Strings

Getting the length of a string

¥To determine how many characters are in a string, you use the built-in `len()` function (short for `length`). The length includes spaces because spaces are characters, each one having a length of one.

```
s1 = ""  
s2 = " "  
s3 = "A B C"  
print(len(s1))  
print(len(s2))  
print(len(s3))
```

Following is the output from that code, when executed

```
0  
1  
5
```

Manipulating Strings

Working with common string operators

Operator	Purpose
<code>x in s</code>	Returns True if <code>x</code> exists somewhere in string <code>s</code> .
<code>x not in s</code>	Returns True if <code>x</code> is not contained in string <code>s</code> .
<code>s * n</code> or <code>n * s</code>	Repeats string <code>s</code> <code>n</code> times.
<code>s[i]</code>	The <code>i</code> th item of string <code>s</code> where the first character is 0.
<code>s[i:j]</code>	A slice from string <code>x</code> beginning with the character at position <code>i</code> through to the character at position <code>j</code> .
<code>s[i:j:k]</code>	A slice of <code>s</code> from <code>i</code> to <code>j</code> with step <code>k</code> .
<code>min(s)</code>	The smallest (lowest) character of string <code>s</code> .
<code>max(s)</code>	The largest (highest) character of string <code>s</code> .
<code>s.index(x[, i[, j]])</code>	The numeric position of the first occurrence of <code>x</code> in string <code>s</code> . The optional <code>i</code> and <code>j</code> limit the search to the characters from <code>i</code> to <code>j</code> .
<code>s.count(x)</code>	The number of times string <code>x</code> appears in larger string <code>s</code> .

FIGURE 1-15:
ASCII numbers
for common
characters.

Number	Character	Number	Character	Number	Character
32	[space]	65	A	97	a
33	!	66	B	98	b
34	"	67	C	99	c
35	#	68	D	100	d
36	\$	69	E	101	e
37	%	70	F	102	f
38	&	71	G	103	g
39	'	72	H	104	h
40	(	73	I	105	i
41	)	74	J	106	j
42	*	75	K	107	k
43	+	76	L	108	l
44	,	77	M	109	m
45	-	78	N	110	n
46	.	79	O	111	o
47	/	80	P	112	p
48	0	81	Q	113	q
49	1	82	R	114	r
50	2	83	S	115	s
51	3	84	T	116	t
52	4	85	U	117	u
53	5	86	V	118	v
54	6	87	W	119	w
55	7	88	X	120	x
56	8	89	Y	121	y
57	9	90	Z	122	z
58	:	91	[	123	{
59	;	92	\	124	
60	<	93	]	125	}
61	=	94	^	126	~
62	>	95	_	127	□
63	?	96	`	128	€
64	@				

Manipulating Strings

Manipulating strings with methods

TABLE 1-5 Built-In Methods for Python 3 Strings

Method	Purpose
<code>s.capitalize()</code>	Returns a string with the first letter capitalized and the rest lowercase.
<code>s.count(x, [y, z])</code>	Returns the number of times string <code>x</code> appears in string <code>s</code> . Optionally, you can add <code>y</code> as a starting point and <code>z</code> as an ending point to search a portion of the string.
<code>s.find(x, [y, z])</code>	Returns a number indicating the first position at which string <code>x</code> can be found in string <code>s</code> . Optional <code>y</code> and <code>z</code> parameters allow you to limit the search to a portion of the string. Returns <code>-1</code> if none found.

<code>s.index(x, [y, z])</code>	Similar to <code>find</code> but returns a “substring not found” error if string <code>x</code> can't be found in string <code>y</code> .
<code>s.isalpha()</code>	Returns <code>True</code> if <code>s</code> is at least one character long and contains only letters (A-Z or a-z).
<code>s.isdecimal()</code>	Returns <code>True</code> if <code>s</code> is at least one character long and contains only numeric characters (0-9).
<code>s.islower()</code>	Returns <code>True</code> if <code>s</code> contains letters and all those letters are lowercase.
<code>s.isnumeric()</code>	Returns <code>True</code> if <code>s</code> is at least one character long and contains only numeric characters (0-9).
<code>s.isprintable()</code>	Returns <code>True</code> if string <code>s</code> contains only printable characters.
<code>s.istitle()</code>	Returns <code>True</code> if string <code>s</code> contains letters and the first letter of each word is uppercase followed by lowercase letters.
<code>s.isupper()</code>	Returns <code>True</code> if all letters in the string are uppercase.
<code>s.lower()</code>	Returns <code>s</code> with all letters converted to lowercase.
<code>s.lstrip()</code>	Returns <code>s</code> with any leading spaces removed.
<code>s.replace(x, y)</code>	Returns a copy of string <code>s</code> with all characters <code>x</code> replaced by character <code>y</code> .
<code>s.rfind(x, [y, z])</code>	Similar to <code>s.find</code> but searches backward from the start of the string. If <code>y</code> and <code>z</code> are provided, searches backward from position <code>z</code> to position <code>y</code> . Returns <code>-1</code> if string <code>x</code> not found.

Method	Purpose
<code>s.rindex()</code>	Same as <code>s.rfind</code> but returns an error if the substring isn't found.
<code>s.rstrip()</code>	Returns string <code>x</code> with any trailing spaces removed.
<code>s.strip()</code>	Returns string <code>x</code> with leading and trailing spaces removed.
<code>s.swapcase()</code>	Returns string <code>s</code> with uppercase letters converted to lowercase and lowercase letters converted to uppercase.
<code>s.title()</code>	Returns string <code>s</code> with the first letter of every word capitalized and all other letters lowercase.
<code>s.upper()</code>	Returns string <code>s</code> with all letters converted to uppercase.



1. Uncovering Dates and Times



Uncovering Dates and Times

- ¥ We often use dates and times for scheduling, or for calculating when something is due or how many days it's past due.
- ¥ We sometimes use timestamps to record exactly when a user did something or when an event occurred.
- ¥ There are lots of reasons for using dates and times in Python, but perhaps surprisingly, no built-in data type for them exists like the ones for strings and numbers.
- ¥ To work with dates and times, you typically need to use the `datetime` module.

Uncovering Dates and Times

Like any module, you must import it before you can use it. You do that using `import datetime`.

As with any import, you can add an alias (nickname) that's easier to type, if you like. For example, import `datetime` as `dt`

- ! `datetime.date` A date consisting of month, day, and year (but no time information)
- ! `datetime.time`: A time consisting of hour, minute, second, microsecond, and optionally time zone information if needed (but no date).
- ! `datetime.datetime` A single item of data consisting of date, time, and optionally time zone information.

Uncovering Dates and Times

Working with dates

- ¥ You can create a date object in two ways. You can get today's date from the computer's internal clock by using the `today()` method. Or you can specify a year, month, and day (in that order) inside parentheses.
- ¥ When specifying the month or day, never use a leading zero for `datetime.date()`. For example, April 1 2020 has to be expressed as `2020,4,1` ~ if you type `2020,04,01`, it won't work.
- ¥ You can isolate any part of a date object by using `.month`, `.day`, or `.year`.

Uncovering Dates and Times

Working with dates

```
print(last_of_teens.month)
print(last_of_teens.day)
print(last_of_teens.year)
```

Following is the output from that code, when executed

```
12
31
2019
```