**“Müasir proqramlaşdırma dilləri (Python for Programmers)” fənni üzrə**

**TƏDRİS-TEMATİK PLAN**

| **№** | **Tarix** | **Mövzular** | **Tədris materialının məzmunu və sərbəst işlər** | **Saat** | **Qeyd** |
| --- | --- | --- | --- | --- | --- |
| **GİRİŞ**  **Müasir proqramlaşdırma dilləri (Python for Programmers) fənninin predmeti, məqsəd və vəzifələri.** | | | | | |
| **I BÖLMƏ. Kompüter hesabı və Python proqramlaşdırma dilinə giriş (1. Introduction to Computers and Python).** | | | | | |
| **Objectives**  In this chapter you’ll:   * Learn about exciting recent developments in computing. * Review objectoriented programming basics. * Understand the strengths of Python. * Be introduced to key Python and datascience libraries you’ll use in this book. * Test-drive the IPython interpreter’s interactive mode for executing Python code. * Execute a Python script that animates a bar chart. * Create and test-drive a web-browser-based Jupyter Notebook for executing Python code. * Learn how big “big data” is and how quickly it’s getting even bigger. * Read a bigdata case study on a popular mobile navigation app. * Be introduced to artificial intelligence—at the intersection of computer science and data science. | | | | | |
| **Outline**  **1.1 Introduction**  **1.2 A Quick Review of Object Technology Basics**  **1.3 Python**  **1.4 It’s the Libraries!**  **1.4.1 Python Standard Library**  **1.4.2 DataScience Libraries**  **1.5 TestDrives: Using IPython and Jupyter Notebooks**  **1.5.1 Using IPython Interactive Mode as a Calculator**  **1.5.2 Executing a Python Program Using the IPython Interpreter**  **1.5.3 Writing and Executing Code in a Jupyter Notebook**  **1.6 The Cloud and the Internet of Things**  **1.6.1 The Cloud**  **1.6.2 Internet of Things**  **1.7 How Big Is Big Data?**  **1.7.1 Big Data Analytics**  **1.7.2 Data Science and Big Data Are Making a Difference: Use Cases**  **1.8 Case Study—A BigData Mobile Application**  **1.9 Intro to Data Science: Artificial Intelligence—at the Intersection of CS and Data Science**  **1.10 WrapUp** | | | | | |
|  |  | **1.1 INTRODUCTION**  Welcome to Python—one of the world’s most widely used computer programming languages and, according to the Popularity of Programming Languages (PYPL) Index, the world’s most popular.  https://pypl.github.io/PYPL.html (as of January 2019). | |  |  |
|  |  | **1.2 A QUICK REVIEW OF OBJECT TECHNOLOGY BASICS**   * **Automobile as an Object** * **Methods and Classes** * **Instantiation** * **Reuse** * **Messages and Method Calls** * **Attributes and Instance Variables** * **Inheritance** * **Object-Oriented Analysis and Design (OOAD)** * **Object-Oriented Programming (OOP)** | |  |  |
|  |  | **1.3 PYTHON**   * **Anaconda Python Distribution**   https://ipython.org/   * **Zen of Python**   https://www.python.org/dev/peps/pep0001/ | |  |  |
|  |  | **1.4 IT’S THE LIBRARIES!**  **1.4.1 Python Standard Library**  **1.4.2 Data-Science Libraries** | |  |  |
|  |  | **1.5 TEST-DRIVES: USING IPYTHON AND JUPYTER NOTEBOOKS**  **1.5.1 Using IPython Interactive Mode as a Calculator**   * **Entering IPython in Interactive Mode** * **Evaluating Expressions** * **Exiting Interactive Mode**   **1.5.2 Executing a Python Program Using the IPython Interpreter**   * **Changing to This Chapter’s Examples Folder** * **Executing the Script** * **Creating Scripts** * **Problems That May Occur at Execution Time**   **1.5.3 Writing and Executing Code in a Jupyter Notebook**   * **Opening JupyterLab in Your Browser** * **Creating a New Jupyter Notebook** * **Renaming the Notebook** * **Evaluating an Expression** * **Adding and Executing Another Cell** * **Saving the Notebook** * **Notebooks Provided with Each Chapter’s Examples** * **Opening and Executing an Existing Notebook** * **Closing JupyterLab** * **JupyterLab Tips**   **More Information on Working with JupyterLab** | |  |  |
|  |  | **1.6 THE CLOUD AND THE INTERNET OF THINGS**  **1.6.1 The Cloud**   * **Mashups**   **1.6.2 Internet of Things** | |  |  |
|  |  | **1.7 HOW BIG IS BIG DATA?**   * **Megabytes (MB)** * **Gigabytes (GB)** * **Terabytes (TB)** * **Petabytes, Exabytes and Zettabytes** * **Additional Big-Data Stats** * **Computing Power Over the Years** * **Processing the World’s Data Requires Lots of Electricity** * **Big-Data Opportunities**   **1.7.1 Big Data Analytics**  **1.7.2 Data Science and Big Data Are Making a Difference: Use Cases** | |  |  |
|  |  | **1.8 CASE STUDY—A BIG-DATA MOBILE APPLICATION** | |  |  |
|  |  | **1.9 INTRO TO DATA SCIENCE: ARTIFICIAL INTELLIGENCE—AT THE INTERSECTION OF CS AND DATA SCIENCE**   * **Artificial-Intelligence Milestones** * **A Personal Anecdote** * **Watson and Big Data Open New Possibilities** * **AI: A Field with Problems But No Solutions** | |  |  |
|  |  | **1.10 WRAP-UP** | |  |  |
| **II BÖLMƏ. Python proqramlaşdırma dilinə giriş (2. Introduction to Python Programming).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Continue using IPython interactive mode to enter code snippets and see their results immediately. * Write simple Python statements and scripts. * Create variables to store data for later use. * Become familiar with builtin data types. * Use arithmetic operators and comparison operators, and understand their precedence. * Use single, double- and triple- quoted strings. * Use built-in function print to display text. * Use built-in function input to prompt the user to enter data at the keyboard and get that data for use in the program. * Convert text to integer values with built-in function int. * Use comparison operators and the if statement to decide whether to execute a statement or group of statements. * Learn about objects and Python’s dynamic typing. * Use built in function type to get an object’s type | | | | | |
| **Outline**  **2.1 Introduction**  **2.2 Variables and Assignment Statements**  **2.3 Arithmetic**  **2.4 Function print and an Intro to Single- and Double- Quoted Strings**  **2.5 Triple- Quoted Strings**  **2.6 Getting Input from the User**  **2.7 Decision Making: The if Statement and Comparison Operators**  **2.8 Objects and Dynamic Typing**  **2.9 Intro to Data Science: Basic Descriptive Statistics**  **2.10 WrapUp** | | | | | |
|  |  | **2.1 INTRODUCTION** | |  |  |
|  |  | **2.2 VARIABLES AND ASSIGNMENT STATEMENTS**   * **Calculations in Assignment Statements** * **Python Style** * **Variable Names** * **Types** | |  |  |
|  |  | **2.3 ARITHMETIC**   * **Multiplication (\*)** * **Exponentiation (\*\*)** * **True Division (/) vs. Floor Division (//)** * **Exceptions and Tracebacks** * **Remainder Operator** * **Straight-Line Form** * **Grouping Expressions with Parentheses** * **Operator Precedence Rules** * **Operator Grouping** * **Redundant Parentheses** * **Operand Types** | |  |  |
|  |  | **2.4 FUNCTION PRINT AND AN INTRO TO SINGLE- AND DOUBLE-QUOTED STRINGS**   * **Printing a Comma-Separated List of Items** * **Printing Many Lines of Text with One Statement** * **Other Escape Sequences** * **Ignoring a Line Break in a Long String** * **Printing the Value of an Expression** | |  |  |
|  |  | **2.5 TRIPLE-QUOTED STRINGS**   * **Including Quotes in Strings** * **Multiline Strings** | |  |  |
|  |  | **2.6 GETTING INPUT FROM THE USER**   * **Function input Always Returns a String** * **Getting an Integer from the User** | |  |  |
|  |  | **2.7 DECISION MAKING: THE IF STATEMENT AND COMPARISON OPERATORS**   * **Making Decisions with the if Statement: Introducing Scripts** * **Comments** * **Docstrings** * **Blank Lines** * **Splitting a Lengthy Statement Across Lines** * **Reading Integer Values from the User** * **if Statements** * **Suite Indentation** * **Confusing == and =** * **Chaining Comparisons** * **Precedence of the Operators We’ve Presented So Far** | |  |  |
|  |  | **2.8 OBJECTS AND DYNAMIC TYPING**   * **Variables Refer to Objects** * **Dynamic Typing** * **Garbage Collection** | |  |  |
|  |  | **2.9 INTRO TO DATA SCIENCE: BASIC DESCRIPTIVE STATISTICS**   * **Determining the Minimum of Three Values** * **Determining the Minimum and Maximum with Built-In Functions min and max** * **Determining the Range of a Collection of Values** * **Functional-Style Programming: Reduction** * **Upcoming Intro to Data Science Sections** | |  |  |
|  |  | **2.10 WRAP-UP** | |  |  |
| **III BÖLMƏ. İdarəetmə operatorları (3. Control Statements).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Make decisions with if, if else and if elif else. * Execute statements repeatedly with while and for. * Shorten assignment expressions with augmented assignments. * Use the for statement and the built-in range function to repeat actions for a sequence of values. * Perform sentinelcontrolled iteration with while. * Create compound conditions with the Boolean operators and, or and not. * Stop looping with break. * Force the next iteration of a loop with continue. * Use functionalstyle programming features to write scripts that are more concise, clearer, easier to debug and easier to parallelize. | | | | | |
| **3.1 Introduction**  **3.2 Control Statements**  **3.3 if Statement**  **3.4 if else and if elif else Statements**  **3.5 while Statement**  **3.6 for Statement**  **3.6.1 Iterables, Lists and Iterators**  **3.6.2 Built-In range Function**  **3.7 Augmented Assignments**  **3.8 Sequence-Controlled Iteration; Formatted Strings**  **3.9 Sentinel-Controlled**  **Iteration**  **3.10 Built-In Function range: A Deeper Look**  **3.11 Using Type Decimal for Monetary Amounts**  **3.12 break and continue Statements**  **3.13 Boolean Operators and, or and not**  **3.14 Intro to Data Science: Measures of Central Tendency—Mean, Median and Mode**  **3.15 Wrap-Up** | | | | | |
|  |  | **3.1 INTRODUCTION** | |  |  |
|  |  | **3.2 CONTROL STATEMENTS**   * **Keywords** | |  |  |
|  |  | **3.3 IF STATEMENT**   * **Suite Indentation** * **Every Expression Can Be Interpreted as Either True or False** * **Confusing == and =** | |  |  |
|  |  | **3.4 IF ELSE AND IF ELIF ELSE STATEMENTS**   * **Conditional Expressions** * **Multiple Statements in a Suite** * **if... elif... else Statement** * **else Is Optional** * **Logic Errors** | |  |  |
|  |  | **3.5 WHILE STATEMENT** | |  |  |
|  |  | **3.6 FOR STATEMENT**   * **Function print’s end Keyword Argument** * **Function print’s sep Keyword Argument**   **3.6.1 Iterables, Lists and Iterators**  **3.6.2 Built-In range Function**   * **Off-By-One Errors** | |  |  |
|  |  | **3.7 AUGMENTED ASSIGNMENTS** | |  |  |
|  |  | **3.8 SEQUENCE-CONTROLLED ITERATION; FORMATTED STRINGS**   * **Introduction to Formatted Strings** | |  |  |
|  |  | **3.9 SENTINEL-CONTROLLED ITERATION**   * **Implementing Sentinel-Controlled Iteration** * **Program Logic for Sentinel-Controlled Iteration** * **Formatting the Class Average with Two Decimal Places** | |  |  |
|  |  | **3.10 BUILT-IN FUNCTION RANGE: A DEEPER LOOK** | |  |  |
|  |  | **3.11 USING TYPE DECIMAL FOR MONETARY AMOUNTS**   * **Importing Type Decimal from the decimal Module** * **Creating Decimals** * **Decimal Arithmetic** * **Compound-Interest Problem Requirements Statement** * **Calculating Compound Interest** * **Formatting the Year and Amount on Deposit** | |  |  |
|  |  | **3.12 BREAK AND CONTINUE STATEMENTS** | |  |  |
|  |  | **3.13 BOOLEAN OPERATORS AND, OR AND NOT**   * **Boolean Operator and** * **Boolean Operator or** * **Improving Performance with Short-Circuit Evaluation** * **Boolean Operator not** | |  |  |
|  |  | **3.14 INTRO TO DATA SCIENCE: MEASURES OF CENTRAL TENDENCY—MEAN, MEDIAN AND MODE** | |  |  |
|  |  | **3.15 WRAP-UP** | |  |  |
| **IV BÖLMƏ. Funksiyalar (4. Functions).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Create custom functions. * Import and use Python Standard Library modules, such as random and math, to reuse code and avoid “reinventing the wheel.” * Pass data between functions. * Generate a range of random numbers. * See simulation techniques using randomnumber generation. * Seed the random number generator to ensure reproducibility. * Pack values into a tuple and unpack values from a tuple. * Return multiple values from a function via a tuple. * Understand how an identifier’s scope determines where in your program you can use it. * Create functions with default parameter values. * Call functions with keyword arguments. * Create functions that can receive any number of arguments. * Use methods of an object. * Write and use a recursive function. | | | | | |
| **Outline**  **4.1 Introduction**  **4.2 Defining Functions**  **4.3 Functions with Multiple Parameters**  **4.4 Random Number Generation**  **4.5 Case Study: A Game of Chance**  **4.6 Python Standard Library**  **4.7 math Module Functions**  **4.8 Using IPython Tab Completion for Discovery**  **4.9 Default Parameter Values**  **4.10 Keyword Arguments**  **4.11 Arbitrary Argument Lists**  **4.12 Methods: Functions That Belong to Objects**  **4.13 Scope Rules**  **4.14 import: A Deeper Look**  **4.15 Passing Arguments to Functions: A Deeper Look**  **4.16 Recursion**  **4.17 Functional Style Programming**  **4.18 Intro to Data Science: Measures of Dispersion**  **4.19 WrapUp** | | | | | |
|  |  | **4.1 INTRODUCTION** | |  |  |
|  |  | **4.2 DEFINING FUNCTIONS**   * **Defining a Custom Function** * **Specifying a Custom Function’s Docstring** * **Returning a Result to a Function’s Caller** * **Local Variables** * **Accessing a Function’s Docstring via IPython’s Help Mechanism** | |  |  |
|  |  | **4.3 FUNCTIONS WITH MULTIPLE PARAMETERS**   * **Function maximum’s Definition** * **Python’s Built-In max and min Functions** | |  |  |
|  |  | **4.4 RANDOM-NUMBER GENERATION**   * **Rolling a Six-Sided Die** * **Rolling a Six-Sided Die 6,000,000 Times** * **Seeding the Random-Number Generator for Reproducibility** | |  |  |
|  |  | **4.5 CASE STUDY: A GAME OF CHANCE**   * **Function roll\_dice—Returning Multiple Values Via a Tuple** * **Function display\_dice** * **First Roll** * **Subsequent Rolls** * **Displaying the Final Results** | |  |  |
|  |  | **4.6 PYTHON STANDARD LIBRARY** | |  |  |
|  |  | **4.7 MATH MODULE FUNCTIONS** | |  |  |
|  |  | **4.8 USING IPYTHON TAB COMPLETION FOR DISCOVERY**   * **Viewing Identifiers in a Module** * **Using the Currently Highlighted Function** | |  |  |
|  |  | **4.9 DEFAULT PARAMETER VALUES** | |  |  |
|  |  | **4.10 KEYWORD ARGUMENTS** | |  |  |
|  |  | **4.11 ARBITRARY ARGUMENT LISTS**   * **Defining a Function with an Arbitrary Argument List** * **Passing an Iterable’s Individual Elements as Function Arguments** | |  |  |
|  |  | **4.12 METHODS: FUNCTIONS THAT BELONG TO OBJECTS** | |  |  |
|  |  | **4.13 SCOPE RULES**   * **Local Scope** * **Global Scope** * **Accessing a Global Variable from a Function** * **Blocks vs. Suites** * **Shadowing Functions** * **Statements at Global Scope** | |  |  |
|  |  | **4.14 IMPORT: A DEEPER LOOK**   * **Importing Multiple Identifiers from a Module** * **Caution: Avoid Wildcard Imports** * **Binding Names for Modules and Module Identifiers** | |  |  |
|  |  | **4.15 PASSING ARGUMENTS TO FUNCTIONS: A DEEPER LOOK**   * **Memory Addresses, References and “Pointers”** * **Built-In Function id and Object Identities** * **Passing an Object to a Function** * **Testing Object Identities with the is Operator** * **Immutable Objects as Arguments** * **Mutable Objects as Arguments** | |  |  |
|  |  | **4.16 RECURSION**   * **Iterative Factorial Approach** * **Recursive Problem Solving** * **Recursive Factorial Approach** * **Visualizing Recursion** * **Implementing a Recursive Factorial Function** * **Indirect Recursion** * **Stack Overflow and Infinite Recursion** | |  |  |
|  |  | **4.17 FUNCTIONAL-STYLE PROGRAMMING**   * **What vs. How** * **Pure Functions** | |  |  |
|  |  | **4.18 INTRO TO DATA SCIENCE: MEASURES OF DISPERSION**   * **Variance** * **Standard Deviation** * **Advantage of Population Standard Deviation vs. Population Variance** | |  |  |
|  |  | **4.19 WRAP-UP** | |  |  |
| **V BÖLMƏ. Ardıcıllıqlar: Siyahılar və Kortejlər (5. Sequences: Lists and Tuples).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Create and initialize lists and tuples. * Refer to elements of lists, tuples and strings. * Sort and search lists, and search tuples. * Pass lists and tuples to functions and methods. * Use list methods to perform common manipulations, such as searching for items, * sorting a list, inserting items and removing items. * Use additional Python functionalstyle * programming capabilities, including lambdas * and the functionalstyle * programming operations filter, map and reduce. * Use functionalstyle * list comprehensions to create lists quickly and easily, and use * generator expressions to generate values on demand. * Use twodimensional * lists. * Enhance your analysis and presentation skills with the Seaborn and Matplotlib * visualization libraries. | | | | | |
| **Outline**  **5.1 Introduction**  **5.2 Lists**  **5.3 Tuples**  **5.4 Unpacking Sequences**  **5.5 Sequence Slicing**  **5.6 del Statement**  **5.7 Passing Lists to Functions**  **5.8 Sorting Lists**  **5.9 Searching Sequences**  **5.10 Other List Methods**  **5.11 Simulating Stacks with Lists**  **5.12 List Comprehensions**  **5.13 Generator Expressions**  **5.14 Filter, Map and Reduce**  **5.15 Other Sequence Processing Functions**  **5.16 TwoDimensional Lists**  **5.17 Intro to Data Science: Simulation and Static Visualizations**   * **5.17.1 Sample Graphs for 600, 60,000 and 6,000,000 Die Rolls** * **5.17.2 Visualizing DieRoll Frequencies and Percentages**   **5.18 WrapUp** | | | | | |
|  |  | **5.1 INTRODUCTION** | |  |  |
|  |  | **5.2 LISTS**   * **Creating a List** * **Accessing Elements of a List** * **Determining a List’s Length** * **Accessing Elements from the End of the List with Negative Indices** * **Indices Must Be Integers or Integer Expressions** * **Lists Are Mutable** * **Some Sequences Are Immutable** * **Attempting to Access a Nonexistent Element** * **Using List Elements in Expressions** * **Appending to a List with +=** * **Concatenating Lists with +** * **Using for and range to Access List Indices and Values** * **Comparison Operators** | |  |  |
|  |  | **5.3 TUPLES**   * **Creating Tuples** * **Accessing Tuple Elements** * **Adding Items to a String or Tuple** * **Appending Tuples to Lists** * **Tuples May Contain Mutable Objects** | |  |  |
|  |  | **5.4 UNPACKING SEQUENCES**   * **Swapping Values Via Packing and Unpacking** * **Accessing Indices and Values Safely with Built-in Function enumerate** * **Creating a Primitive Bar Chart** | |  |  |
|  |  | **5.5 SEQUENCE SLICING**   * **Specifying a Slice with Starting and Ending Indices** * **Specifying a Slice with Only an Ending Index** * **Specifying a Slice with Only a Starting Index** * **Specifying a Slice with No Indices** * **Slicing with Steps** * **Slicing with Negative Indices and Steps** * **Modifying Lists Via Slices** | |  |  |
|  |  | **5.6 DEL STATEMENT**   * **Deleting the Element at a Specific List Index** * **Deleting a Slice from a List** * **Deleting a Slice Representing the Entire List** * **Deleting a Variable from the Current Session** | |  |  |
|  |  | **5.7 PASSING LISTS TO FUNCTIONS**   * **Passing an Entire List to a Function** * **Passing a Tuple to a Function** * **A Note Regarding Tracebacks** | |  |  |
|  |  | **5.8 SORTING LISTS**   * **Sorting a List in Ascending Order** * **Sorting a List in Descending Order** * **Built-In Function sorted** | |  |  |
|  |  | **5.9 SEARCHING SEQUENCES**   * **List Method index** * **Specifying the Starting Index of a Search** * **Specifying the Starting and Ending Indices of a Search** * **Operators in and not in** * **Using Operator in to Prevent a ValueError** * **Built-In Functions any and all** | |  |  |
|  |  | **5.10 OTHER LIST METHODS**   * **Inserting an Element at a Specific List Index** * **Adding an Element to the End of a List** * **Adding All the Elements of a Sequence to the End of a List** * **Removing the First Occurrence of an Element in a List** * **Emptying a List** * **Counting the Number of Occurrences of an Item** * **Reversing a List’s Elements** * **Copying a List** | |  |  |
|  |  | **5.11 SIMULATING STACKS WITH LISTS** | |  |  |
|  |  | **5.12 LIST COMPREHENSIONS**   * **Using a List Comprehension to Create a List of Integers** * **Mapping: Performing Operations in a List Comprehension’s Expression** * **Filtering: List Comprehensions with if Clauses** * **List Comprehension That Processes Another List’s Elements** | |  |  |
|  |  | **5.13 GENERATOR EXPRESSIONS** | |  |  |
|  |  | **5.14 FILTER, MAP AND REDUCE**   * **Filtering a Sequence’s Values with the Built-In filter Function** * **Using a lambda Rather than a Function** * **Mapping a Sequence’s Values to New Values** * **Combining filter and map** * **Reduction: Totaling the Elements of a Sequence with sum** | |  |  |
|  |  | **5.15 OTHER SEQUENCE PROCESSING FUNCTIONS**   * **Finding the Minimum and Maximum Values Using a Key Function** * **Iterating Backward Through a Sequence** * **Combining Iterables into Tuples of Corresponding Elements** | |  |  |
|  |  | **5.16 TWO-DIMENSIONAL LISTS**   * **Creating a Two-Dimensional List** * **Illustrating a Two-Dimensional List** * **Identifying the Elements in a Two-Dimensional List** * **How the Nested Loops Execute** | |  |  |
|  |  | **5.17 INTRO TO DATA SCIENCE: SIMULATION AND STATIC VISUALIZATIONS**  **5.17.1 Sample Graphs for 600, 60,000 and 6,000,000 Die Rolls**  **5.17.2 Visualizing Die-Roll Frequencies and Percentages**   * **Launching IPython for Interactive Matplotlib Development** * **Importing the Libraries** * **Rolling the Die and Calculating Die Frequencies** * **Creating the Initial Bar Plot** * **Setting the Window Title and Labeling the x- and y-Axes** * **Finalizing the Bar Plot** * **Rolling Again and Updating the Bar Plot—Introducing IPython Magics** * **Saving Snippets to a File with the %save Magic** * **Command-Line Arguments; Displaying a Plot from a Script** | |  |  |
|  |  | **5.18 WRAP-UP** | |  |  |
| **VI BÖLMƏ. Lüğətlər və Çoxluqlar (6. Dictionaries and Sets).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Use dictionaries to represent unordered collections of key–value pairs. * Use sets to represent unordered collections of unique values. * Create, initialize and refer to elements of dictionaries and sets. * Iterate through a dictionary’s keys, values and key–value pairs. * Add, remove and update a dictionary’s key–value pairs. * Use dictionary and set comparison operators. * Combine sets with set operators and methods. * Use operators in and not in to determine if a dictionary contains a key or a set * contains a value. * Use the mutable set operations to modify a set’s contents. * Use comprehensions to create dictionaries and sets quickly and conveniently. * Learn how to build dynamic visualizations. * Enhance your understanding of mutability and immutability. | | | | | |
| **Outline**  **6.1 Introduction**  **6.2 Dictionaries**  **6.2.1 Creating a Dictionary**  **6.2.2 Iterating through a Dictionary**  **6.2.3 Basic Dictionary Operations**  **6.2.4 Dictionary Methods keys and values**  **6.2.5 Dictionary Comparisons**  **6.2.6 Example: Dictionary of Student Grades**  **6.2.7 Example: Word Counts**  **6.2.8 Dictionary Method update**  **6.2.9 Dictionary Comprehensions**  **6.3 Sets**  **6.3.1 Comparing Sets**  **6.3.2 Mathematical Set Operations**  **6.3.3 Mutable Set Operators and Methods**  **6.3.4 Set Comprehensions**  **6.4 Intro to Data Science: Dynamic Visualizations**  **6.4.1 How Dynamic Visualization Works**  **6.4.2 Implementing a Dynamic Visualization**  **6.5 WrapUp** | | | | | |
|  |  | **6.1 INTRODUCTION** | |  |  |
|  |  | **6.2 DICTIONARIES**   * **Unique Keys**   **6.2.1 Creating a Dictionary**   * **Determining if a Dictionary Is Empty**   **6.2.2 Iterating through a Dictionary**  **6.2.3 Basic Dictionary Operations**   * **Accessing the Value Associated with a Key** * **Updating the Value of an Existing Key–Value Pair** * **Adding a New Key–Value Pair** * **Removing a Key–Value Pair** * **Attempting to Access a Nonexistent Key** * **Testing Whether a Dictionary Contains a Specified Key**   **6.2.4 Dictionary Methods keys and values**   * **Dictionary Views** * **Converting Dictionary Keys, Values and Key–Value Pairs to Lists** * **Processing Keys in Sorted Order**   **6.2.5 Dictionary Comparisons**  **6.2.6 Example: Dictionary of Student Grades**  **6.2.7 Example: Word Counts**   * **Python Standard Library Module collections**   **6.2.8 Dictionary Method update**  **6.2.9 Dictionary Comprehensions** | |  |  |
|  |  | **6.3 SETS**   * **Creating a Set with Curly Braces** * **Determining a Set’s Length** * **Checking Whether a Value Is in a Set** * **Iterating Through a Set** * **Creating a Set with the Built-In set Function** * **Frozenset: An Immutable Set Type**   **6.3.1 Comparing Sets**  **6.3.2 Mathematical Set Operations**   * **Union** * **Intersection** * **Difference** * **Symmetric Difference** * **Disjoint**   **6.3.3 Mutable Set Operators and Methods**   * **Mutable Mathematical Set Operations** * **Methods for Adding and Removing Elements**   **6.3.4 Set Comprehensions** | |  |  |
|  |  | **6.4 INTRO TO DATA SCIENCE: DYNAMIC VISUALIZATIONS**   * **The Law of Large Numbers**   **6.4.1 How Dynamic Visualization Works**   * **Animation Frames** * **Running RollDieDynamic.py** * **Sample Executions**   **6.4.2 Implementing a Dynamic Visualization**   * **Importing the Matplotlib animation Module** * **Function update** * **Function update: Rolling the Die and Updating the frequencies List** * **Function update: Configuring the Bar Plot and Text** * **Variables Used to Configure the Graph and Maintain State** * **Calling the animation Module’s FuncAnimation Function** | |  |  |
|  |  | **6.5 WRAP-UP** | |  |  |
| **VII BÖLMƏ. Massiv – NumPy Obyekt-yönlü Proqramlaşdırması (7. Array-Oriented Programming with NumPy).** | | | | | |
| **Objectives**  **In this chapter you’ll:**   * Learn how arrays differ from lists. * Use the numpy module’s highperformance ndarrays. * Compare list and ndarray performance with the IPython %timeit magic. * Use ndarrays to store and retrieve data efficiently. * Create and initialize ndarrays. * Refer to individual ndarray elements. * Iterate through ndarrays. * Create and manipulate multidimensional ndarrays. * Perform common ndarray manipulations. * Create and manipulate pandas onedimensional Series and twodimensional DataFrames. * Customize Series and DataFrame indices. * Calculate basic descriptive statistics for data in a Series and a DataFrame. * Customize floatingpoint number precision in pandas output formatting. | | | | | |
| **Outline**  **7.1 Introduction**  **7.2 Creating arrays from Existing Data**  **7.3 array Attributes**  **7.4 Filling arrays with Specific Values**  **7.5 Creating arrays from Ranges**  **7.6 List vs. array Performance: Introducing %timeit**  **7.7 array Operators**  **7.8 NumPy Calculation Methods**  **7.9 Universal Functions**  **7.10 Indexing and Slicing**  **7.11 Views: Shallow Copies**  **7.12 Deep Copies**  **7.13 Reshaping and Transposing**  **7.14 Intro to Data Science: pandas Series and DataFrames**  **7.14.1 pandas Series**  **7.14.2 DataFrames**  **7.15 WrapUp** | | | | | |
|  |  | **7.1 INTRODUCTION** | |  |  |
|  |  | **7.2 CREATING ARRAYS FROM EXISTING DATA**   * **Multidimensional Arguments** | |  |  |
|  |  | **7.3 ARRAY ATTRIBUTES**   * **Determining an array’s Element Type** * **Determining an array’s Dimensions** * **Determining an array’s Number of Elements and Element Size** * **Iterating Through a Multidimensional array’s Elements** | |  |  |
|  |  | **7.4 FILLING ARRAYS WITH SPECIFIC VALUES** | |  |  |
|  |  | **7.5 CREATING ARRAYS FROM RANGES**   * **Creating Integer Ranges with arange** * **Creating Floating-Point Ranges with linspace** * **Reshaping an array** * **Displaying Large arrays** | |  |  |
|  |  | **7.6 LIST VS. ARRAY PERFORMANCE: INTRODUCING %TIMEIT**   * **Timing the Creation of a List Containing Results of 6,000,000 Die Rolls** * **Timing the Creation of an array Containing Results of 6,000,000 Die Rolls** * **60,000,000 and 600,000,000 Die Rolls** * **Customizing the %timeit Iterations** * **Other IPython Magics** | |  |  |
|  |  | **7.7 ARRAY OPERATORS**   * **Arithmetic Operations with arrays and Individual Numeric Values** * **Broadcasting** * **Arithmetic Operations Between arrays** * **Comparing arrays** | |  |  |
|  |  | **7.8 NUMPY CALCULATION METHODS**   * **Calculations by Row or Column** | |  |  |
|  |  | **7.9 UNIVERSAL FUNCTIONS**   * **Broadcasting with Universal Functions** * **Other Universal Functions** | |  |  |
|  |  | **7.10 INDEXING AND SLICING**   * **Indexing with Two-Dimensional arrays** * **Selecting a Subset of a Two-Dimensional array’s Rows** * **Selecting a Subset of a Two-Dimensional array’s Columns** | |  |  |
|  |  | **7.11 VIEWS: SHALLOW COPIES**   * **Slice Views** | |  |  |
|  |  | **7.12 DEEP COPIES**   * **Module copy—Shallow vs. Deep Copies for Other Types of Python Objects** | |  |  |
|  |  | **7.13 RESHAPING AND TRANSPOSING**   * **flatten vs. ravel** * **Transposing Rows and Columns** * **Horizontal and Vertical Stacking** * **Dictionary Initializers** * **Accessing Elements of a Series Via Custom Indices** * **Creating a Series of Strings** | |  |  |
|  |  | **7.14 INTRO TO DATA SCIENCE: PANDAS SERIES AND DATAFRAMES**  **7.14.1 pandas Series**   * **Creating a Series with Default Indices** * **Displaying a Series** * **Creating a Series with All Elements Having the Same Value** * **Accessing a Series’ Elements** * **Producing Descriptive Statistics for a Series** * **Creating a Series with Custom Indices**   **7.14.2 DataFrames**   * **Creating a DataFrame from a Dictionary** * **Customizing a DataFrame’s Indices with the index Attribute** * **Accessing a DataFrame’s Columns** * **Selecting Rows via the loc and iloc Attributes** * **Selecting Rows via Slices and Lists with the loc and iloc Attributes** * **Selecting Subsets of the Rows and Columns** * **Boolean Indexing** * **Accessing a Specific DataFrame Cell by Row and Column** * **Descriptive Statistics** * **Transposing the DataFrame with the T Attribute** * **Sorting by Rows by Their Indices** * **Sorting by Column Indices** * **Sorting by Column Values** * **Copy vs. In-Place Sorting** | |  |  |
|  |  | **7.15 WRAP-UP** | |  |  |
| **VIII BÖLMƏ. Sətirlər: Daha dərin baxış (8. Strings: A Deeper Look).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Understand text processing. * Use string methods. * Format string content. * Concatenate and repeat strings. * Strip whitespace from the ends of strings. * Change characters from lowercase to uppercase and vice versa. * Compare strings with the comparison operators. * Search strings for substrings and replace substrings. * Split strings into tokens. * Concatenate strings into a single string with a specified separator between items. * Create and use regular expressions to match patterns in strings, replace substrings and validate data. * Use regular expression metacharacters, quantifiers, character classes and grouping. * Understand how critical string manipulations are to natural language processing. * Understand the data science terms data munging, data wrangling and data cleaning, and use regular expressions to munge data into preferred formats. | | | | | |
| **Outline**  **8.1 Introduction**  **8.2 Formatting Strings**  **8.2.1 Presentation Types**  **8.2.2 Field Widths and Alignment**  **8.2.3 Numeric Formatting**  **8.2.4 String’s format Method**  **8.3 Concatenating and Repeating Strings**  **8.4 Stripping Whitespace from Strings**  **8.5 Changing Character Case**  **8.6 Comparison Operators for Strings**  **8.7 Searching for Substrings**  **8.8 Replacing Substrings**  **8.9 Splitting and Joining Strings**  **8.10 Characters and CharacterTesting Methods**  **8.11 Raw Strings**  **8.12 Introduction to Regular Expressions**  **8.12.1 re Module and Function fullmatch**  **8.12.2 Replacing Substrings and Splitting Strings**  **8.12.3 Other Search Functions; Accessing Matches**  **8.13 Intro to Data Science: Pandas, Regular Expressions and Data Munging**  **8.14 WrapUp** | | | | | |
|  |  | **8.1 INTRODUCTION** | |  |  |
|  |  | **8.2 FORMATTING STRINGS**  **8.2.1 Presentation Types**   * **Integers** * **Characters** * **Strings** * **Floating-Point and Decimal Values**   **8.2.2 Field Widths and Alignment**   * **Explicitly Specifying Left and Right Alignment in a Field** * **Centering a Value in a Field**   **8.2.3 Numeric Formatting**   * **Formatting Positive Numbers with Signs** * **Using a Space Where a + Sign Would Appear in a Positive Value** * **Grouping Digits**   **8.2.4 String’s format Method**   * **Multiple Placeholders** * **Referencing Arguments By Position Number** * **Referencing Keyword Arguments** | |  |  |
|  |  | **8.3 CONCATENATING AND REPEATING STRINGS** | |  |  |
|  |  | **8.4 STRIPPING WHITESPACE FROM STRINGS**   * **Removing Leading and Trailing Whitespace** * **Removing Leading Whitespace** * **Removing Trailing Whitespace** | |  |  |
|  |  | **8.5 CHANGING CHARACTER CASE**   * **Capitalizing Only a String’s First Character** * **Capitalizing the First Character of Every Word in a String** | |  |  |
|  |  | **8.6 COMPARISON OPERATORS FOR STRINGS** | |  |  |
|  |  | **8.7 SEARCHING FOR SUBSTRINGS**   * **Counting Occurrences** * **Locating a Substring in a String** * **Determining Whether a String Contains a Substring** * **Locating a Substring at the Beginning or End of a String** | |  |  |
|  |  | **8.8 REPLACING SUBSTRINGS** | |  |  |
|  |  | **8.9 SPLITTING AND JOINING STRINGS**   * **Splitting Strings** * **Joining Strings** * **String Methods partition and rpartition** * **String Method splitlines** | |  |  |
|  |  | **8.10 CHARACTERS AND CHARACTER-TESTING METHODS** | |  |  |
|  |  | **8.11 RAW STRINGS** | |  |  |
|  |  | **8.12 INTRODUCTION TO REGULAR EXPRESSIONS**   * **Validating Data** * **Other Uses of Regular Expressions**   **8.12.1 re Module and Function fullmatch**   * **Matching Literal Characters** * **Metacharacters, Character Classes and Quantifiers** * **Other Predefined Character Classes** * **Custom Character Classes** * **vs. + Quantifier** * **Other Quantifiers**   **8.12.2 Replacing Substrings and Splitting Strings**   * **Function sub—Replacing Patterns** * **Function split**   **8.12.3 Other Search Functions; Accessing Matches**   * **Function search—Finding the First Match Anywhere in a String** * **Ignoring Case with the Optional flags Keyword Argument** * **Metacharacters That Restrict Matches to the Beginning or End of a String** * **Function findall and finditer—Finding All Matches in a String** * **Capturing Substrings in a Match** | |  |  |
|  |  | **8.13 INTRO TO DATA SCIENCE: PANDAS, REGULAR EXPRESSIONS AND DATA MUNGING**   * **Cleaning Your Data** * **Data Validation** * **Reformatting Your Data** | |  |  |
|  |  | **8.14 WRAP-UP** | |  |  |
| **IX BÖLMƏ. Fayllar və İstisnalar (9. Files and Exceptions).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Understand the notions of files and persistent data. * Read, write and update files. * Read and write CSV files, a common format for machinelearning datasets. * Serialize objects into the JSON datainterchange format—commonly used to transmit over the Internet—and deserialize JSON into objects. * Use the with statement to ensure that resources are properly released, avoiding “resource leaks”. * Use the try statement to delimit code in which exceptions may occur and handle those exceptions with associated except clauses. * Use the try statement’s else clause to execute code when no exceptions occur in the try suite. * Use the try statement’s finally clause to execute code regardless of whether an exception occurs in the try. * raise exceptions to indicate runtime problems. * Understand the traceback of functions and methods that led to an exception. * Use pandas to load into a DataFrame and process the Titanic Disaster CSV dataset. | | | | | |
| **Outline**  **9.1 Introduction**  **9.2 Files**  **9.3 TextFile Processing**  **9.3.1 Writing to a Text File: Introducing the with Statement**  **9.3.2 Reading Data from a Text File**  **9.4 Updating Text Files**  **9.5 Serialization with JSON**  **9.6 Focus on Security: pickle Serialization and Deserialization**  **9.7 Additional Notes Regarding Files**  **9.8 Handling Exceptions**  **9.8.1 Division by Zero and Invalid Input**  **9.8.2 try Statements**  **9.8.3 Catching Multiple Exceptions in One except Clause**  **9.8.4 What Exceptions Does a Function or Method Raise?**  **9.8.5 What Code Should Be Placed in a try Suite?**  **9.9 finally Clause**  **9.10 Explicitly Raising an Exception**  **9.11 (Optional) Stack Unwinding and Tracebacks**  **9.12 Intro to Data Science: Working with CSV Files**  **9.12.1 Python Standard Library Module csv**  **9.12.2 Reading CSV Files into Pandas DataFrames**  **9.12.3 Reading the Titanic Disaster Dataset**  **9.12.4 Simple Data Analysis with the Titanic Disaster Dataset**  **9.12.5 Passenger Age Histogram**  **9.13 WrapUp** | | | | | |
|  |  | **9.1 INTRODUCTION** | |  |  |
|  |  | **9.2 FILES**   * **End of File** * **Standard File Objects** | |  |  |
|  |  | **9.3 TEXT-FILE PROCESSING**  **9.3.1 Writing to a Text File: Introducing the with Statement**   * **The with Statement** * **Built-In Function open** * **Writing to the File** * **Contents of accounts.txt File**   **9.3.2 Reading Data from a Text File**   * **File Method readlines** * **Seeking to a Specific File Position** | |  |  |
|  |  | **9.4 UPDATING TEXT FILES**   * **Updating accounts.txt** * **os Module File-Processing Functions** | |  |  |
|  |  | **9.5 SERIALIZATION WITH JSON**   * **JSON Data Format** * **Python Standard Library Module json** * **Serializing an Object to JSON** * **Deserializing the JSON Text** * **Displaying the JSON Text** | |  |  |
|  |  | **9.6 FOCUS ON SECURITY: PICKLE SERIALIZATION AND DESERIALIZATION** | |  |  |
|  |  | **9.7 ADDITIONAL NOTES REGARDING FILES**   * **Other File Object Methods** | |  |  |
|  |  | **9.8 HANDLING EXCEPTIONS**  **9.8.1 Division by Zero and Invalid Input**   * **Division By Zero** * **Invalid Input**   **9.8.2 try Statements**   * **try Clause** * **except Clause** * **else Clause** * **Flow of Control for a ZeroDivisionError** * **Flow of Control for a ValueError** * **Flow of Control for a Successful Division**   **9.8.3 Catching Multiple Exceptions in One except Clause**  **9.8.4 What Exceptions Does a Function or Method Raise?**  **9.8.5 What Code Should Be Placed in a try Suite?** | |  |  |
|  |  | **9.9 FINALLY CLAUSE**   * **The finally Clause of the try Statement** * **Example** * **Combining with Statements and try except Statements** | |  |  |
|  |  | **9.10 EXPLICITLY RAISING AN EXCEPTION** | |  |  |
|  |  | **9.11 (OPTIONAL) STACK UNWINDING AND TRACEBACKS**   * **Traceback Details** * **Stack Unwinding** * **Tip for Reading Tracebacks** * **Exceptions in finally Suites** | |  |  |
|  |  | **9.12 INTRO TO DATA SCIENCE: WORKING WITH CSV FILES**  **9.12.1 Python Standard Library Module csv**   * **Writing to a CSV File** * **Reading from a CSV File** * **Caution: Commas in CSV Data Fields** * **Caution: Missing Commas and Extra Commas in CSV Files**   **9.12.2 Reading CSV Files into Pandas DataFrames**   * **Datasets** * **Working with Locally Stored CSV Files**   **9.12.3 Reading the Titanic Disaster Dataset**   * **Loading the Titanic Dataset via a URL** * **Viewing Some of the Rows in the Titanic Dataset** * **Customizing the Column Names**   **9.12.4 Simple Data Analysis with the Titanic Disaster Dataset**  **9.12.5 Passenger Age Histogram** | |  |  |
|  |  | **9.13 WRAP-UP** | |  |  |
| **X BÖLMƏ. Obyekt-Yönlü Proqramlaşdırma (10. Object-Oriented Programming).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Create custom classes and objects of those classes. * Understand the benefits of crafting valuable classes. * Control access to attributes. * Appreciate the value of object orientation. * Use Python special methods \_\_repr\_\_, \_\_str\_\_ and \_\_format\_\_ to get an object’s string representations. * Use Python special methods to overload (redefine) operators to use them with objects of new classes. * Inherit methods, properties and attributes from existing classes into new classes, then customize those classes. * Understand the inheritance notions of base classes (superclasses) and derived classes (subclasses). * Understand duck typing and polymorphism that enable “programming in the general.” * Understand class object from which all classes inherit fundamental capabilities. * Compare composition and inheritance. * Build test cases into docstrings and run these tests with doctest, Understand namespaces and how they affect scope. | | | | | |
| **Outline**  **10.1 Introduction**  **10.2 Custom Class Account**  **10.2.1 TestDriving Class Account**  **10.2.2 Account Class Definition**  **10.2.3 Composition: Object References as Members of Classes**  **10.3 Controlling Access to Attributes**  **10.4 Properties for Data Access**  **10.4.1 TestDriving Class Time**  **10.4.2 Class Time Definition**  **10.4.3 Class Time Definition Design Notes**  **10.5 Simulating “Private” Attributes**  **10.6 Case Study: Card Shuffling and Dealing Simulation**  **10.6.1 TestDriving Classes Card and DeckOfCards**  **10.6.2 Class Card—Introducing Class Attributes**  **10.6.3 Class DeckOfCards**  **10.6.4 Displaying Card Images with Matplotlib**  **10.7 Inheritance: Base Classes and Subclasses**  **10.8 Building an Inheritance Hierarchy; Introducing Polymorphism**  **10.8.1 Base Class CommissionEmployee**  **10.8.2 Subclass SalariedCommissionEmployee**  **10.8.3 Processing CommissionEmployees and SalariedCommissionEmployees Polymorphically**  **10.8.4 A Note About ObjectBased and ObjectOriented Programming**  **10.9 Duck Typing and Polymorphism**  **10.10 Operator Overloading**  **10.10.1 TestDriving Class Complex**  **10.10.2 Class Complex Definition**  **10.11 Exception Class Hierarchy and Custom Exceptions**  **10.12 Named Tuples**  **10.13 A Brief Intro to Python 3.7’s New Data Classes**  **10.13.1 Creating a Card Data Class**  **10.13.2 Using the Card Data Class**  **10.13.3 Data Class Advantages over Named Tuples**  **10.13.4 Data Class Advantages over Traditional Classes**  **10.14 Unit Testing with Docstrings and doctest**  **10.15 Namespaces and Scopes**  **10.16 Intro to Data Science: Time Series and Simple Linear Regression**  **10.17 WrapUp** | | | | | |
|  |  | **10.1 INTRODUCTION**   * **Crafting Valuable Classes** * **Class Libraries and Object-Based Programming** * **Creating Your Own Custom Classes** * **Inheritance** * **Polymorphism** * **An Entertaining Case Study: Card-Shuffling-and-Dealing Simulation** * **Data Classes** * **Other Concepts Introduced in This Chapter** | |  |  |
|  |  | **10.2 CUSTOM CLASS ACCOUNT**  **10.2.1 Test-Driving Class Account**   * **Importing Classes Account and Decimal** * **Create an Account Object with a Constructor Expression** * **Getting an Account’s Name and Balance** * **Depositing Money into an Account** * **Account Methods Perform Validation**   **10.2.2 Account Class Definition**   * **Defining a Class** * **Initializing Account Objects: Method \_\_init\_\_** * **Method deposit** | |  |  |
|  |  | **10.3 CONTROLLING ACCESS TO ATTRIBUTES**   * **Encapsulation** * **Leading Underscore (\_) Naming Convention** | |  |  |
|  |  | **10.4 PROPERTIES FOR DATA ACCESS**  **10.4.1 Test-Driving Class Time**   * **Creating a Time Object** * **Displaying a Time Object** * **Getting an Attribute Via a Property** * **Setting the Time** * **Setting an Attribute via a Property** * **Attempting to Set an Invalid Value**   **10.4.2 Class Time Definition**   * **Class Time: \_\_init\_\_ Method with Default Parameter Values** * **Class Time: hour Read-Write Property** * **Class Time: minute and second Read-Write Properties** * **Class Time: Method set\_time** * **Class Time: Special Method \_\_repr\_\_** * **Class Time: Special Method \_\_str\_\_**   **10.4.3 Class Time Definition Design Notes**   * **Interface of a Class** * **Attributes Are Always Accessible** * **Internal Data Representation** * **Evolving a Class’s Implementation Details** * **Properties** * **Utility Methods** * **Module datetime** | |  |  |
|  |  | **10.5 SIMULATING “PRIVATE” ATTRIBUTES**   * **IPython Auto-Completion Shows Only “Public” Attributes** * **Demonstrating “Private” Attributes** | |  |  |
|  |  | **10.6 CASE STUDY: CARD SHUFFLING AND DEALING SIMULATION**  **10.6.1 Test-Driving Classes Card and DeckOfCards**   * **Creating, Shuffling and Dealing the Cards** * **Dealing Cards** * **Class Card’s Other Features** * **10.6.2 Class Card—Introducing Class Attributes** * **Class Attributes FACES and SUITS** * **Card Method \_\_init\_\_** * **Read-Only Properties face, suit and image\_name** * **Methods That Return String Representations of a Card**   **10.6.3 Class DeckOfCards**   * **Method \_\_init\_\_** * **Method shuffle** * **Method deal\_card** * **Method \_\_str\_\_**   **10.6.4 Displaying Card Images with Matplotlib**   * **Enable Matplotlib in IPython** * **Create the Base Path for Each Image** * **Import the Matplotlib Features** * **Create the Figure and Axes Objects** * **Configure the Axes Objects and Display the Images** * **Maximize the Image Sizes** * **Shuffle and Re-Deal the Deck** | |  |  |
|  |  | **10.7 INHERITANCE: BASE CLASSES AND SUBCLASSES**   * **CommunityMember Inheritance Hierarchy** * **Shape Inheritance Hierarchy** | |  |  |
|  |  | **10.8 BUILDING AN INHERITANCE HIERARCHY; INTRODUCING POLYMORPHISM**  **10.8.1 Base Class CommissionEmployee**   * **All Classes Inherit Directly or Indirectly from Class object** * **Testing Class CommissionEmployee**   **10.8.2 Subclass SalariedCommissionEmployee**   * **Declaring Class SalariedCommissionEmployee** * **Inheriting from Class CommissionEmployee** * **Method \_\_init\_\_ and Built-In Function super** * **Overriding Method earnings** * **Overriding Method \_\_repr\_\_** * **Testing Class SalariedCommissionEmployee** * **Testing the “is a” Relationship**   **10.8.3 Processing CommissionEmployees and SalariedCommissionEmployees Polymorphically**  **10.8.4 A Note About Object-Based and Object-Oriented Programming** | |  |  |
|  |  | **10.9 DUCK TYPING AND POLYMORPHISM** | |  |  |
|  |  | **10.10 OPERATOR OVERLOADING**   * **Operator Overloading Restrictions** * **Complex Numbers**   **10.10.1 Test-Driving Class Complex**  **10.10.2 Class Complex Definition**   * **Method \_\_init\_\_** * **Overloaded + Operator** * **Overloaded += Augmented Assignment** * **Method \_\_repr\_\_** | |  |  |
|  |  | **10.11 EXCEPTION CLASS HIERARCHY AND CUSTOM EXCEPTIONS**   * **Catching Base-Class Exceptions** * **Custom Exception Classes** | |  |  |
|  |  | **10.12 NAMED TUPLES** | |  |  |
|  |  | **10.13 A BRIEF INTRO TO PYTHON 3.7’S NEW DATA CLASSES**   * **Data Classes Autogenerate Code**   **10.13.1 Creating a Card Data Class**   * **Importing from the dataclasses and typing Modules** * **Using the @dataclass Decorator** * **Variable Annotations: Class Attributes** * **Variable Annotations: Data Attributes** * **Defining a Property and Other Methods** * **Variable Annotation Notes**   **10.13.2 Using the Card Data Class**  **10.13.3 Data Class Advantages over Named Tuples**  **10.13.4 Data Class Advantages over Traditional Classes**   * **More Information** | |  |  |
|  |  | **10.14 UNIT TESTING WITH DOCSTRINGS AND DOCTEST**   * **Module doctest and the testmod Function** * **Modified Account Class** * **Module \_\_main\_\_** * **Running Tests** * **IPython %doctest\_mode Magic** | |  |  |
|  |  | **10.15 NAMESPACES AND SCOPES**   * **Local Namespace** * **Global Namespace** * **Built-In Namespace** * **Finding Identifiers in Namespaces** * **Nested Functions** * **Class Namespace** * **Object Namespace** | |  |  |
|  |  | **10.16 INTRO TO DATA SCIENCE: TIME SERIES AND SIMPLE LINEAR REGRESSION**   * **Time Series** * **Simple Linear Regression** * **Linear Relationships** * **Components of the Simple Linear Regression Equation** * **Function linregress from the SciPy’s stats Module** * **Pandas** * **Seaborn Visualization** * **Getting Weather Data from NOAA** * **Loading the Average High Temperatures into a DataFrame** * **Cleaning the Data** * **Calculating Basic Descriptive Statistics for the Dataset** * **Forecasting Future January Average High Temperatures** * **Plotting the Average High Temperatures and a Regression Line** * **Getting Time Series Datasets** | |  |  |
|  |  | **10.17 WRAP-UP** | |  |  |
| **XI BÖLMƏ. Təbii dillərin emalı (11. Natural Language Processing (NLP)).** | | | | | |
| **Objectives**  **In this chapter you’ll:**   * Perform natural language processing (NLP) tasks, which are fundamental to many of the forthcoming data science case study chapters. * Run lots of NLP demos. * Use the TextBlob, NLTK, Textatistic and spaCy NLP libraries and their pretrained models to perform various NLP tasks. * Tokenize text into words and sentences. * Use partsofspeech tagging. * Use sentiment analysis to determine whether text is positive, negative or neutral. * Detect the language of text and translate between languages using TextBlob’s Google Translate support. * Get word roots via stemming and lemmatization. * Use TextBlob’s spell checking and correction capabilities. * Get word definitions, synonyms and antonyms. * Remove stop words from text. * Create word clouds. * Determine text readability with Textatistic. * Use the spaCy library for named entity recognition and similarity detection. | | | | | |
| **Outline**  **11.1 Introduction**  **11.2 TextBlob**  **11.2.1 Create a TextBlob**  **11.2.2 Tokenizing Text into Sentences and Words**  **11.2.3 PartsofSpeech Tagging**  **11.2.4 Extracting Noun Phrases**  **11.2.5 Sentiment Analysis with TextBlob’s Default Sentiment Analyzer**  **11.2.6 Sentiment Analysis with the NaiveBayesAnalyzer**  **11.2.7 Language Detection and Translation**  **11.2.8 Inflection: Pluralization and Singularization**  **11.2.9 Spell Checking and Correction**  **11.2.10 Normalization: Stemming and Lemmatization**  **11.2.11 Word Frequencies**  **11.2.12 Getting Definitions, Synonyms and Antonyms from WordNet**  **11.2.13 Deleting Stop Words**  **11.2.14 ngrams**  **11.3 Visualizing Word Frequencies with Bar Charts and Word Clouds**  **11.3.1 Visualizing Word Frequencies with Pandas**  **11.3.2 Visualizing Word Frequencies with Word Clouds**  **11.4 Readability Assessment with Textatistic**  **11.5 Named Entity Recognition with spaCy**  **11.6 Similarity Detection with spaCy**  **11.7 Other NLP Libraries and Tools**  **11.8 Machine Learning and Deep Learning Natural Language Applications**  **11.9 Natural Language Datasets**  **11.10 WrapUp** | | | | | |
|  |  | **11.1 INTRODUCTION** | |  |  |
|  |  | **11.2 TEXTBLOB**   * **Installing the TextBlob Module** * **Project Gutenberg**   **11.2.1 Create a TextBlob**  **11.2.2 Tokenizing Text into Sentences and Words**  **11.2.3 Parts-of-Speech Tagging**  **11.2.4 Extracting Noun Phrases**  **11.2.5 Sentiment Analysis with TextBlob’s Default Sentiment Analyzer**   * **Getting the Sentiment of a TextBlob** * **Getting the polarity and subjectivity from the Sentiment Object** * **Getting the Sentiment of a Sentence**   **11.2.6 Sentiment Analysis with the NaiveBayesAnalyzer**  **11.2.7 Language Detection and Translation**  **11.2.8 Inflection: Pluralization and Singularization**  **11.2.9 Spell Checking and Correction**  **11.2.10 Normalization: Stemming and Lemmatization**  **11.2.11 Word Frequencies**  **11.2.12 Getting Definitions, Synonyms and Antonyms from WordNet**   * **Getting Definitions** * **Getting Synonyms** * **Getting Antonyms**   **11.2.13 Deleting Stop Words**  **11.2.14 n-grams** | |  |  |
|  |  | **11.3 VISUALIZING WORD FREQUENCIES WITH BAR CHARTS AND WORD CLOUDS**  **11.3.1 Visualizing Word Frequencies with Pandas**   * **Loading the Data** * **Getting the Word Frequencies** * **Eliminating the Stop Words** * **Sorting the Words by Frequency** * **Getting the Top 20 Words** * **Convert top20 to a DataFrame** * **Visualizing the DataFrame**   **11.3.2 Visualizing Word Frequencies with Word Clouds**   * **Installing the wordcloud Module** * **Loading the Text** * **Loading the Mask Image that Specifies the Word Cloud’s Shape** * **Configuring the WordCloud Object** * **Generating the Word Cloud** * **Saving the Word Cloud as an Image File** * **Generating a Word Cloud from a Dictionary** * **Displaying the Image with Matplotlib** | |  |  |
|  |  | **11.4 READABILITY ASSESSMENT WITH TEXTATISTIC**   * **Install Textatistic** * **Calculating Statistics and Readability Scores** | |  |  |
|  |  | **11.5 NAMED ENTITY RECOGNITION WITH SPACY**   * **Install spaCy** * **Loading the Language Model** * **Creating a spaCy Doc** * **Getting the Named Entities** | |  |  |
|  |  | **11.6 SIMILARITY DETECTION WITH SPACY**   * **Loading the Language Model and Creating a spaCy Doc** * **Creating the spaCy Docs** * **Comparing the Books’ Similarity** | |  |  |
|  |  | **11.7 OTHER NLP LIBRARIES AND TOOLS** | |  |  |
|  |  | **11.8 MACHINE LEARNING AND DEEP LEARNING NATURAL LANGUAGE APPLICATIONS** | |  |  |
|  |  | **11.9 NATURAL LANGUAGE DATASETS** | |  |  |
|  |  | **11.10 WRAP-UP** | |  |  |
| **XII BÖLMƏ. Verilənlərin İntellektual Analizi (12. Data Mining Twitter).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * Understand Twitter’s impact on businesses, brands, reputation, sentiment analysis, predictions and more. * Use Tweepy, one of the most popular Python Twitter API clients for data mining Twitter. * Use the Twitter Search API to download past tweets that meet your criteria. * Use the Twitter Streaming API to sample the stream of live tweets as they’re happening. * See that the tweet objects returned by Twitter contain valuable information beyond the tweet text. * Use the natural language processing techniques from the last chapter to clean and preprocess tweets to prepare them for analysis. * Perform sentiment analysis on tweets. * Spot trends with Twitter’s Trends API. * Map tweets using folium and OpenStreetMap. * Understand various ways to store tweets using techniques discussed throughout this book. | | | | | |
| **Outline**  **12.1 Introduction**  **12.2 Overview of the Twitter APIs**  **12.3 Creating a Twitter Account**  **12.4 Getting Twitter Credentials—Creating an App**  **12.5 What’s in a Tweet?**  **12.6 Tweepy**  **12.7 Authenticating with Twitter Via Tweepy**  **12.8 Getting Information About a Twitter Account**  **12.9 Introduction to Tweepy Cursors: Getting an Account’s Followers and Friends**  **12.9.1 Determining an Account’s Followers**  **12.9.2 Determining Whom an Account Follows**  **12.9.3 Getting a User’s Recent Tweets**  **12.10 Searching Recent Tweets**  **12.11 Spotting Trends: Twitter Trends API**  **12.11.1 Places with Trending Topics**  **12.11.2 Getting a List of Trending Topics**  **12.11.3 Create a Word Cloud from Trending Topics**  **12.12 Cleaning/Preprocessing Tweets for Analysis**  **12.13 Twitter Streaming API**  **12.13.1 Creating a Subclass of StreamListener**  **12.13.2 Initiating Stream Processing**  **12.14 Tweet Sentiment Analysis**  **12.15 Geocoding and Mapping**  **12.15.1 Getting and Mapping the Tweets**  **12.15.2 Utility Functions in tweetutilities.py**  **12.15.3 Class LocationListener**  **12.16 Ways to Store Tweets**  **12.17 Twitter and Time Series**  **12.18 WrapUp** | | | | | |
|  |  | **12.1 INTRODUCTION**   * **What Is Twitter?** * **Twitter Statistics** * **Twitter and Big Data** * **Cautions** | |  |  |
|  |  | **12.2 OVERVIEW OF THE TWITTER APIS**   * **Rate Limits: A Word of Caution** * **Other Restrictions** | |  |  |
|  |  | **12.3 CREATING A TWITTER ACCOUNT** | |  |  |
|  |  | **12.4 GETTING TWITTER CREDENTIALS—CREATING AN APP**   * **Getting Your Credentials** * **Storing Your Credentials** * **OAuth 2.0** | |  |  |
|  |  | **12.5 WHAT’S IN A TWEET?**   * **Key Properties of a Tweet Object** * **Sample Tweet JSON** * **Twitter JSON Object Resources** | |  |  |
|  |  | **12.6 TWEEPY**   * **Installing Tweepy** * **Installing geopy** | |  |  |
|  |  | **12.7 AUTHENTICATING WITH TWITTER VIA TWEEPY**   * **Creating and Configuring an OAuthHandler to Authenticate with Twitter** * **Creating an API Object** | |  |  |
|  |  | **12.8 GETTING INFORMATION ABOUT A TWITTER ACCOUNT**   * **Getting Basic Account Information** * **Getting the Most Recent Status Update** * **Getting the Number of Followers** * **Getting the Number of Friends** * **Getting Your Own Account’s Information** | |  |  |
|  |  | **12.9 INTRODUCTION TO TWEEPY CURSORS: GETTING AN ACCOUNT’S FOLLOWERS AND FRIENDS**  **12.9.1 Determining an Account’s Followers**   * **Creating a Cursor** * **Getting Results** * **Automatic Paging** * **Getting Follower IDs Rather Than Followers**   **12.9.2 Determining Whom an Account Follows**  **12.9.3 Getting a User’s Recent Tweets**   * **Grabbing Recent Tweets from Your Own Timeline** | |  |  |
|  |  | **12.10 SEARCHING RECENT TWEETS**   * **Utility Function print\_tweets from tweetutilities.py** * **Searching for Specific Words** * **Searching with Twitter Search Operators** * **Searching for a Hashtag** | |  |  |
|  |  | **12.11 SPOTTING TRENDS: TWITTER TRENDS API**  **12.11.1 Places with Trending Topics**  **12.11.2 Getting a List of Trending Topics**   * **Worldwide Trending Topics** * **New York City Trending Topics**   **12.11.3 Create a Word Cloud from Trending Topics** | |  |  |
|  |  | **12.12 CLEANING/PREPROCESSING TWEETS FOR ANALYSIS**   * **tweet-preprocessor Library and TextBlob Utility Functions** * **Installing tweet-preprocessor** * **Cleaning a Tweet** | |  |  |
|  |  | **12.13 TWITTER STREAMING API**  **12.13.1 Creating a Subclass of StreamListener**   * **Class TweetListener** * **Class TweetListener: \_\_init\_\_ Method** * **Class TweetListener: on\_connect Method** * **Class TweetListener: on\_status Method**   **12.13.2 Initiating Stream Processing**   * **Authenticating** * **Creating a TweetListener** * **Creating a Stream** * **Starting the Tweet Stream** * **Asynchronous vs. Synchronous Streams** * **Other filter Method Parameters** * **Twitter Restrictions Note** | |  |  |
|  |  | **12.14 TWEET SENTIMENT ANALYSIS**   * **Imports** * **Class SentimentListener: \_\_init\_\_ Method** * **Method on\_status** * **Main Application** | |  |  |
|  |  | **12.15 GEOCODING AND MAPPING**   * **geopy Library** * **OpenMapQuest Geocoding API** * **Folium Library and Leaflet.js JavaScript Mapping Library** * **Maps from OpenStreetMap.org**   **12.15.1 Getting and Mapping the Tweets**   * **Get the API Object** * **Collections Required By LocationListener** * **Creating the LocationListener** * **Configure and Start the Stream of Tweets** * **Displaying the Location Statistics** * **Geocoding the Locations** * **Displaying the Bad Location Statistics** * **Cleaning the Data** * **Creating a Map with Folium** * **Creating Popup Markers for the Tweet Locations** * **Saving the Map**   **12.15.2 Utility Functions in tweetutilities.py**   * **get\_tweet\_content Utility Function** * **get\_geocodes Utility Function** * **12.15.3 Class LocationListener** | |  |  |
|  |  | **12.16 WAYS TO STORE TWEETS** | |  |  |
|  |  | **12.17 TWITTER AND TIME SERIES** | |  |  |
|  |  | **12.18 WRAP-UP** | |  |  |
| **XIII BÖLMƏ. 13. IBM Watson və Koqnitiv Hesablamalar (13. IBM Watson and Cognitive Computing).** | | | | | |
| **Objectives**  **In this chapter, you’ll:**   * See Watson’s range of services and use their Lite tier to become familiar with them at no charge. * Try lots of demos of Watson services. * Understand what cognitive computing is and how you can incorporate it into your applications. * Register for an IBM Cloud account and get credentials to use various services. * Install the Watson Developer Cloud Python SDK to interact with Watson services. * Develop a traveler’s companion language translator app by using Python to weave together a mashup of the Watson Speech to Text, Language Translator and Text to Speech services. * Check out additional resources, such as IBM Watson Redbooks that will help you jump start your custom Watson application development. | | | | | |
| **Outline**  **13.1 Introduction: IBM Watson and Cognitive Computing**  **13.2 IBM Cloud Account and Cloud Console**  **13.3 Watson Services**  **13.4 Additional Services and Tools**  **13.5 Watson Developer Cloud Python SDK**  **13.6 Case Study: Traveler’s Companion Translation App**  **13.6.1 Before You Run the App**  **13.6.2 TestDriving the App**  **13.6.3 SimpleLanguageTranslator.py Script Walkthrough**  **13.7 Watson Resources**  **13.8 WrapUp** | | | | | |
|  |  | **13.1 INTRODUCTION: IBM WATSON AND COGNITIVE COMPUTING** | |  |  |
|  |  | **13.2 IBM CLOUD ACCOUNT AND CLOUD CONSOLE** | |  |  |
|  |  | **13.3 WATSON SERVICES**   * **Watson Assistant** * **Visual Recognition** * **Speech to Text** * **Text to Speech** * **Language Translator** * **Natural Language Understanding** * **Discovery** * **Personality Insights** * **Tone Analyzer** * **Natural Language Classifier** * **Synchronous and Asynchronous Capabilities** | |  |  |
|  |  | **13.4 ADDITIONAL SERVICES AND TOOLS**   * **Watson Studio** * **Knowledge Studio** * **Machine Learning** * **Knowledge Catalog** * **Cognos Analytics** | |  |  |
|  |  | **13.5 WATSON DEVELOPER CLOUD PYTHON SDK**   * **Modules We’ll Need for Audio Recording and Playback** * **SDK Examples** | |  |  |
|  |  | **13.6 CASE STUDY: TRAVELER’S COMPANION TRANSLATION APP**  **13.6.1 Before You Run the App**   * **Registering for the Speech to Text Service** * **Registering for the Text to Speech Service** * **Registering for the Language Translator Service** * **Retrieving Your Credentials**   **13.6.2 Test-Driving the App**   * **Processing the Question** * **Processing the Response**   **13.6.3 SimpleLanguageTranslator.py Script Walkthrough**   * **Importing Watson SDK Classes** * **Other Imported Modules** * **Main Program: Function run\_translator** * **Function translate** * **Function text\_to\_speech** * **Function record\_audio** * **Function play\_audio** * **Executing the run\_translator Function** | |  |  |
|  |  | **13.7 WATSON RESOURCES**   * **Watson Services Documentation** * **Watson SDKs** * **Learning Resources** * **Watson Videos** * **IBM Redbooks** | |  |  |
|  |  | **13.8 WRAP-UP** | |  |  |
| **XIV BÖLMƏ. Maşın öyrənməsi: təsnifat, reqressiya və klasterləşmə (14. Machine Learning: Classification, Regression and Clustering).** | | | | | |
| **Objectives**  **In this chapter you’ll:**   * Use scikitlearn with popular datasets to perform machine learning studies. * Use Seaborn and Matplotlib to visualize and explore data. * Perform supervised machine learning with knearest neighbors classification and linear regression. * Perform multiclassification with Digits dataset. * Divide a dataset into training, test and validation sets. * Tune model hyperparameters with kfold crossvalidation. * Measure model performance. * Display a confusion matrix showing classification prediction hits and misses. * Perform multiple linear regression with the California Housing dataset. * Perform dimensionality reduction with PCA and tSNE on the Iris and Digits datasets to prepare them for twodimensional visualizations. * Perform unsupervised machine learning with kmeans clustering and the Iris dataset. | | | | | |
| **Outline**  **14.1 Introduction to Machine Learning**  **14.1.1 ScikitLearn**  **14.1.2 Types of Machine Learning**  **14.1.3 Datasets Bundled with ScikitLearn**  **14.1.4 Steps in a Typical Data Science Study**  **14.2 Case Study: Classification with kNearest Neighbors and the Digits Dataset, Part 1**  **14.2.1 kNearest Neighbors Algorithm**  **14.2.2 Loading the Dataset**  **14.2.3 Visualizing the Data**  **14.2.4 Splitting the Data for Training and Testing**  **14.2.5 Creating the Model**  **14.2.6 Training the Model**  **14.2.7 Predicting Digit Classes**  **14.3 Case Study: Classification with kNearest Neighbors and the Digits Dataset, Part 2**  **14.3.1 Metrics for Model Accuracy**  **14.3.2 Kfold CrossValidation**  **14.3.3 Running Multiple Models to Find the Best One**  **14.3.4 Hyperparameter Tuning**  **14.4 Case Study: Time Series and Simple Linear Regression**  **14.5 Case Study: Multiple Linear Regression with the California Housing Dataset**  **14.5.1 Loading the Dataset**  **14.5.2 Exploring the Data with Pandas**  **14.5.3 Visualizing the Feature**  **14.5.4 Splitting the Data for Training and Testing**  **14.5.5 Training the Model**  **14.5.6 Testing the Model**  **14.5.7 Visualizing the Expected vs. Predicted Prices**  **14.5.8 Regression Model Metrics**  **14.5.9 Choosing the Best Model**  **14.6 Case Study: Unsupervised Machine Learning, Part 1—Dimensionality Reduction**  **14.7 Case Study: Unsupervised Machine Learning, Part 2—kMeans Clustering**  **14.7.1 Loading the Iris Dataset**  **14.7.2 Exploring the Iris Dataset: Descriptive Statistics with Pandas**  **14.7.3 Visualizing the Dataset with a Seaborn pairplot**  **14.7.4 Using a KMeans Estimator**  **14.7.5 Dimensionality Reduction with Principal Component Analysis**  **14.7.6 Choosing the Best Clustering Estimator**  **14.8 WrapUp** | | | | | |
|  |  | **14.1 INTRODUCTION TO MACHINE LEARNING**   * **What Is Machine Learning?** * **Prediction** * **Machine Learning Applications**   **14.1.1 Scikit-Learn**   * **Which Scikit-Learn Estimator Should You Choose for Your Project**   **14.1.2 Types of Machine Learning**   * **Supervised Machine Learning** * **Datasets** * **Classification** * **Regression** * **Unsupervised Machine Learning** * **K-Means Clustering and the Iris Dataset** * **Big Data and Big Computer Processing Power**   **14.1.3 Datasets Bundled with Scikit-Learn**  **14.1.4 Steps in a Typical Data Science Study** | |  |  |
|  |  | **14.2 CASE STUDY: CLASSIFICATION WITH K-NEAREST NEIGHBORS AND THE DIGITS DATASET, PART 1**   * **Classification Problems** * **Our Approach**   **14.2.1 k-Nearest Neighbors Algorithm**   * **Hyperparameters and Hyperparameter Tuning**   **14.2.2 Loading the Dataset**   * **Displaying the Description** * **Checking the Sample and Target Sizes** * **A Sample Digit Image** * **Preparing the Data for Use with Scikit-Learn** * **Creating the Diagram** * **Displaying Each Image and Removing the Axes Labels**   **14.2.4 Splitting the Data for Training and Testing**   * **Training and Testing Set Sizes**   **14.2.5 Creating the Model**  **14.2.6 Training the Model**  **14.2.7 Predicting Digit Classes** | |  |  |
|  |  | **14.3 CASE STUDY: CLASSIFICATION WITH K-NEAREST NEIGHBORS AND THE DIGITS DATASET, PART 2**   * **Estimator Method score** * **Confusion Matrix** * **Classification Report** * **Visualizing the Confusion Matrix**   **14.3.2 K-Fold Cross-Validation**   * **KFold Class** * **Using the KFold Object with Function cross\_val\_score**   **14.3.3 Running Multiple Models to Find the Best One**   * **Scikit-Learn Estimator Diagram**   **14.3.4 Hyperparameter Tuning** | |  |  |
|  |  | **14.4 CASE STUDY: TIME SERIES AND SIMPLE LINEAR REGRESSION**   * **Loading the Average High Temperatures into a DataFrame** * **Splitting the Data for Training and Testing** * **Training the Model** * **Testing the Model** * **Predicting Future Temperatures and Estimating Past Temperatures** * **Visualizing the Dataset with the Regression Line** * **Overfitting/Underfitting** | |  |  |
|  |  | **14.5 CASE STUDY: MULTIPLE LINEAR REGRESSION WITH THE CALIFORNIA HOUSING DATASET**  **14.5.1 Loading the Dataset**   * **Loading the Data** * **Displaying the Dataset’s Description**   **14.5.2 Exploring the Data with Pandas**  **14.5.3 Visualizing the Features**  **14.5.4 Splitting the Data for Training and Testing**  **14.5.5 Training the Model**  **14.5.6 Testing the Model**  **14.5.7 Visualizing the Expected vs. Predicted Prices**  **14.5.8 Regression Model Metrics**  **14.5.9 Choosing the Best Model** | |  |  |
|  |  | **14.6 CASE STUDY: UNSUPERVISED MACHINE LEARNING, PART 1—DIMENSIONALITY REDUCTION**   * **Loading the Digits Dataset** * **Creating a TSNE Estimator for Dimensionality Reduction** * **Transforming the Digits Dataset’s Features into Two Dimensions** * **Visualizing the Reduced Data** * **Visualizing the Reduced Data with Different Colors for Each Digit** | |  |  |
|  |  | **14.7 CASE STUDY: UNSUPERVISED MACHINE LEARNING, PART 2—K-MEANS CLUSTERING**   * **Iris Dataset**   **14.7.1 Loading the Iris Dataset**   * **Checking the Numbers of Samples, Features and Targets**   **14.7.2 Exploring the Iris Dataset: Descriptive Statistics with Pandas**  **14.7.3 Visualizing the Dataset with a Seaborn pairplot**   * **Displaying the pairplot in One Color**   **14.7.4 Using a KMeans Estimator**   * **Creating the Estimator** * **Fitting the Model** * **Comparing the Computer Cluster Labels to the Iris Dataset’s Target Values**   **14.7.5 Dimensionality Reduction with Principal Component**   * **Analysis** * **Creating the PCA Object** * **Transforming the Iris Dataset’s Features into Two Dimensions** * **Visualizing the Reduced Data**   **14.7.6 Choosing the Best Clustering Estimator** | |  |  |
|  |  | **14.8 WRAP-UP** | |  |  |
| **XV BÖLMƏ. Dərin Öyrənmə (15. Deep Learning).** | | | | | |
| **Objectives**  **In this chapter you’ll:**   * Understand what a neural network is and how it enables deep learning. * Create Keras neural networks. * Understand Keras layers, activation functions, loss functions and optimizers. * Use a Keras convolutional neural network (CNN) trained on the MNIST dataset to recognize handwritten digits. * Use a Keras recurrent neural network (RNN) trained on the IMDb dataset to perform binary classification of positive and negative movie reviews. * Use TensorBoard to visualize the progress of training deeplearning networks. * Learn which pretrained neural networks come with Keras. * Understand the value of using models pretrained on the massive ImageNet dataset for computer vision apps. | | | | | |
| **Outline**  **15.1 Introduction**  **15.1.1 Deep Learning Applications**  **15.1.2 Deep Learning Demos**  **15.1.3 Keras Resources**  **15.2 Keras BuiltIn Datasets**  **15.3 Custom Anaconda Environments**  **15.4 Neural Networks**  **15.5 Tensors**  **15.6 Convolutional Neural Networks for Vision; MultiClassification with the MNIST Dataset**  **15.6.1 Loading the MNIST Dataset**  **15.6.2 Data Exploration**  **15.6.3 Data Preparation**  **15.6.4 Creating the Neural Network**  **15.6.5 Training and Evaluating the Model**  **15.6.6 Saving and Loading a Model**  **15.7 Visualizing Neural Network Training with TensorBoard**  **15.8 ConvnetJS: BrowserBased DeepLearning Training and Visualization**  **15.9 Recurrent Neural Networks for Sequences; Sentiment Analysis with the IMDb Dataset**  **15.9.1 Loading the IMDb Movie Reviews Dataset**  **15.9.2 Data Exploration**  **15.9.3 Data Preparation**  **15.9.4 Creating the Neural Network**  **15.9.5 Training and Evaluating the Model**  **15.10 Tuning Deep Learning Models**  **15.11Convnet Models Pretrained on ImageNet**  **15.12 Wrap-Up** | | | | | |
|  |  | **15.1 INTRODUCTION**   * **Keras and TensorFlow** * **Models** * **Experiment with Your Models** * **Dataset Sizes** * **Processing Power** * **Bundled Datasets** * **Future of Deep Learning**   **15.1.1 Deep Learning Applications**  **15.1.2 Deep Learning Demos**  **15.1.3 Keras Resources** | |  |  |
|  |  | **15.2 KERAS BUILT-IN DATASETS** | |  |  |
|  |  | **15.3 CUSTOM ANACONDA ENVIRONMENTS**   * **Environments in Anaconda** * **Creating an Anaconda Environment** * **Activating an Alternate Anaconda Environment** * **Deactivating an Alternate Anaconda Environment** * **Jupyter Notebooks and JupyterLab** | |  |  |
|  |  | **15.4 NEURAL NETWORKS**   * **Artificial Neurons** * **Artificial Neural Network Diagram** * **Learning Is an Iterative Process** * **How Artificial Neurons Decide Whether to Activate Synapses** | |  |  |
|  |  | **15.5 TENSORS**   * **High-Performance Processors** | |  |  |
|  |  | **15.6 CONVOLUTIONAL NEURAL NETWORKS FOR VISION; MULTI-CLASSIFICATION WITH THE MNIST DATASET**   * **Reproducibility in Keras and Deep Learning** * **Basic Keras Neural Network** * **Launch JupyterLab**   **15.6.1 Loading the MNIST Dataset**   * **Visualizing Digits**   **15.6.3 Data Preparation**   * **Reshaping the Image Data** * **Normalizing the Image Data** * **One-Hot Encoding: Converting the Labels From Integers to Categorical Data**   **15.6.4 Creating the Neural Network**   * **Adding Layers to the Network** * **Convolution** * **Adding a Convolution Layer** * **Dimensionality of the First Convolution Layer’s Output** * **Overfitting** * **Adding a Pooling Layer** * **Adding Another Convolutional Layer and Pooling Layer** * **Flattening the Results** * **Adding a Dense Layer to Reduce the Number of Features** * **Adding Another Dense Layer to Produce the Final Output** * **Printing the Model’s Summary** * **Visualizing a Model’s Structure** * **Compiling the Model**   **15.6.5 Training and Evaluating the Model**   * **Evaluating the Model** * **Making Predictions** * **Locating the Incorrect Predictions** * **Visualizing Incorrect Predictions** * **Displaying the Probabilities for Several Incorrect Predictions**   **15.6.6 Saving and Loading a Model** | |  |  |
|  |  | **15.7 VISUALIZING NEURAL NETWORK TRAINING WITH TENSORBOARD**   * **Executing TensorBoard** * **The TensorBoard Dashboard** * **Copy the MNIST Convnet’s Notebook** * **Configuring Keras to Write the TensorBoard Log Files** * **Updating Our Call to fit** | |  |  |
|  |  | **15.8 CONVNETJS: BROWSER-BASED DEEP-LEARNING TRAINING AND VISUALIZATION**   * **Training Stats** * **Instantiate a Network and Trainer** * **Network Visualization** * **Example Predictions on Test Set** | |  |  |
|  |  | **15.9 RECURRENT NEURAL NETWORKS FOR SEQUENCES; SENTIMENT ANALYSIS WITH THE IMDB DATASET**  **15.9.1 Loading the IMDb Movie Reviews Dataset**  **15.9.2 Data Exploration**   * **Movie Review Encodings** * **Decoding a Movie Review**   **15.9.3 Data Preparation**   * **Splitting the Test Data into Validation and Test Data**   **15.9.4 Creating the Neural Network**   * **Adding an Embedding Layer** * **Adding an LSTM Layer** * **Adding a Dense Output Layer** * **Compiling the Model and Displaying the Summary**   **15.9.5 Training and Evaluating the Model** | |  |  |
|  |  | **15.10 TUNING DEEP LEARNING MODELS** | |  |  |
|  |  | **15.11 CONVNET MODELS PRETRAINED ON IMAGENET**   * **Keras Pretrained Convnet Models** * **Reusing Pretrained Models** * **ImageNet Challenge** | |  |  |
|  |  | **15.12 WRAP-UP** | |  |  |
| **XVI BÖLMƏ. 16. Böyük Verilənlər: Hadoop, Spark, NoSQL və IoT (16. Big Data: Hadoop, Spark, NoSQL and IoT).** | | | | | |
| **Objectives**  **In this chapter you’ll:**   * Understand what big data is and how quickly it’s getting bigger. * Manipulate a SQLite relational database using Structured Query Language (SQL). * Understand the four major types of NoSQL databases. * Store tweets in a MongoDB NoSQL JSON document database and visualize them on a Folium map. * Understand Apache Hadoop and how it’s used in bigdata batchprocessing applications. * Build a Hadoop MapReduce application on Microsoft’s Azure HDInsight cloud service. * Understand Apache Spark and how it’s used in highperformance, realtime bigdata applications. * Use Spark streaming to process data in minibatches. * Understand the Internet of Things (IoT) and the publish/subscribe model. * Publish messages from a simulated Internetconnected device and visualize its messages in a dashboard. * Subscribe to PubNub’s live Twitter and IoT streams and visualize the data. | | | | | |
| **Outline**  **16.1 Introduction**  **16.2 Relational Databases and Structured Query Language (SQL)**  **16.3.1 A books Database**  **16.3.2 SELECT Queries**  **16.3.3 WHERE Clause**  **16.3.4 ORDER BY Clause**  **16.3.5 Merging Data from Multiple Tables: INNER JOIN**  **16.3.6 INSERT INTO Statement**  **16.3.7 UPDATE Statement**  **16.3.8 DELETE FROM Statement**  **16.3 NoSQL and NewSQL BigData Databases: A Brief Tour**  **16.3.1 NoSQL Key–Value Databases**  **16.3.2 NoSQL Document Databases**  **16.3.3 NoSQL Columnar Databases**  **16.3.4 NoSQL Graph Databases**  **16.3.5 NewSQL Databases**  **16.4 Case Study: A MongoDB JSON Document Database**  **16.4.1 Creating the MongoDB Atlas Cluster**  **16.4.2 Streaming Tweets into MongoDB**  **16.5 Hadoop**  **16.5.1 Hadoop Overview**  **16.6.2 Summarizing Word Lengths in Romeo and Juliet via MapReduce**  **16.5.3 Creating an Apache Hadoop Cluster in Microsoft Azure HDInsight**  **16.5.4 Hadoop Streaming**  **16.5.5 Implementing the Mapper**  **16.5.6 Implementing the Reducer**  **16.5.7 Preparing to Run the MapReduce Example**  **16.5.8 Running the MapReduce Job**  **16.6 Spark**  **16.6.1 Spark Overview**  **16.6.2 Docker and the Jupyter Docker Stacks**  **16.6.3 Word Count with Spark**  **16.6.4 Spark Word Count on Microsoft Azure**  **16.7 Spark Streaming: Counting Twitter Hashtags Using the pyspark-notebook Docker Stack**  **16.7.1 Streaming Tweets to a Socket**  **16.7.2 Summarizing Tweet Hashtags; Introducing Spark SQL**  **16.8 Internet of Things and Dashboards**  **16.8.1 Publish and Subscribe**  **16.8.2 Visualizing a PubNub Sample Live Stream with a Freeboard Dashboard**  **16.8.3 Simulating an InternetConnected Thermostat in Python**  **16.8.4 Creating the Dashboard with Freeboard.io**  **16.8.5 Creating a Python PubNub Subscriber**  **16.9 WrapUp** | | | | | |
|  |  | **16.1 INTRODUCTION**   * **Databases** * **Apache Hadoop** * **Apache Spark** * **Internet of Things** * **Experience Cloud and Desktop Big-Data Software** * **Algorithms and Data** * **Data’s Meaning** * **Big-Data Sources** | |  |  |
|  |  | **16.2 RELATIONAL DATABASES AND STRUCTURED QUERY LANGUAGE (SQL)**   * **Tables, Rows and Columns** * **Selecting Data Subsets** * **SQLite**   **16.2.1 A books Database**   * **Creating the books Database** * **Connecting to the Database in Python** * **authors Table** * **Viewing the authors Table’s Contents** * **titles Table** * **author\_ISBN Table** * **Entity-Relationship (ER) Diagram** * **SQL Keywords**   **16.2.2 SELECT Queries**  **16.2.3 WHERE Clause**   * **Pattern Matching: Zero or More Characters** * **Pattern Matching: Any Character**   **16.2.4 ORDER BY Clause**   * **Sorting By Multiple Columns** * **Combining the WHERE and ORDER BY Clauses**   **16.2.5 Merging Data from Multiple Tables: INNER JOIN**  **16.2.6 INSERT INTO Statement**  **16.2.7 UPDATE Statement**  **16.2.8 DELETE FROM Statement**   * **Closing the Database** * **SQL in Big Data** | |  |  |
|  |  | **16.3 NOSQL AND NEWSQL BIG-DATA DATABASES: A BRIEF TOUR**  **16.3.1 NoSQL Key–Value Databases**  **16.3.2 NoSQL Document Databases**  **16.3.3 NoSQL Columnar Databases**  **16.3.4 NoSQL Graph Databases**  **16.3.5 NewSQL Databases** | |  |  |
|  |  | **16.4 CASE STUDY: A MONGODB JSON DOCUMENT DATABASE**   * **Installing the Python Libraries Required for Interacting with MongoDB** * **keys.py**   **16.4.1 Creating the MongoDB Atlas Cluster**   * **Creating Your First Database User** * **Whitelist Your IP Address** * **Connect to Your Cluster**   **16.4.2 Streaming Tweets into MongoDB**   * **Use Tweepy to Authenticate with Twitter** * **Loading the Senators’ Data** * **Configuring the MongoClient** * **Setting up Tweet Stream** * **Starting the Tweet Stream** * **Class TweetListener** * **Counting Tweets for Each Senator** * **Show Tweet Counts for Each Senator** * **Get the State Locations for Plotting Markers** * **Grouping the Tweet Counts by State** * **Creating the Map** * **Creating a Choropleth to Color the Map** * **Creating the Map Markers for Each State** * **Displaying the Map** | |  |  |
|  |  | **16.5 HADOOP**  **16.5.1 Hadoop Overview**   * **HDFS, MapReduce and YARN** * **Hadoop Ecosystem** * **Hadoop Providers** * **Hadoop 3**   **16.5.2 Summarizing Word Lengths in Romeo and Juliet via MapReduce**  **16.5.3 Creating an Apache Hadoop Cluster in Microsoft Azure HDInsight**   * **Creating an HDInsight Hadoop Cluster**   **16.5.4 Hadoop Streaming**  **16.5.5 Implementing the Mapper**  **16.5.6 Implementing the Reducer**  **16.5.7 Preparing to Run the MapReduce Example**   * **Copying the Script Files to the HDInsight Hadoop Cluster** * **Copying RomeoAndJuliet into the Hadoop File System**   **16.5.8 Running the MapReduce Job**   * **Viewing the Word Counts** * **Deleting Your Cluster So You Do Not Incur Charges** | |  |  |
|  |  | **16.6 SPARK**  **16.6.1 Spark Overview**   * **History** * **Architecture and Components** * **Providers**   **16.6.2 Docker and the Jupyter Docker Stacks**   * **Installing Docker** * **Jupyter Docker Stacks** * **Run Jupyter Docker Stack** * **Opening JupyterLab in Your Browser** * **Accessing the Docker Container’s Command Line** * **Stopping and Restarting a Docker Container**   **16.6.3 Word Count with Spark**   * **Loading the NLTK Stop Words** * **Configuring a SparkContext** * **Reading the Text File and Mapping It to Words** * **Removing the Stop Words** * **Counting Each Remaining Word** * **Locating Words with Counts Greater Than or Equal to 60** * **Sorting and Displaying the Results**   **16.6.4 Spark Word Count on Microsoft Azure**   * **Create an Apache Spark Cluster in HDInsight Using the Azure Portal** * **Install Libraries into a Cluster** * **Copying RomeoAndJuliet.txt to the HDInsight Cluster** * **Accessing Jupyter Notebooks in HDInsight** * **Uploading the RomeoAndJulietCounter.ipynb Notebook** * **Modifying the Notebook to Work with Azure** | |  |  |
|  |  | **16.7 SPARK STREAMING: COUNTING TWITTER HASHTAGS USING THE PYSPARK-NOTEBOOK DOCKER STACK**   * **Launching the Docker Container and Installing Tweepy**   **16.7.1 Streaming Tweets to a Socket**   * **Executing the Script in the Docker Container** * **starttweetstream.py import Statements** * **Class TweetListener** * **Main Application**   **16.7.2 Summarizing Tweet Hashtags; Introducing Spark SQL**   * **Importing the Libraries** * **Utility Function to Get the SparkSession** * **Utility Function to Display a Barchart Based on a Spark DataFrame** * **Utility Function to Summarize the Top-20 Hashtags So Far** * **Getting the SparkContext** * **Getting the StreamingContext** * **Setting Up a Checkpoint for Maintaining State** * **Connecting to the Stream via a Socket** * **Tokenizing the Lines of Hashtags** * **Mapping the Hashtags to Tuples of Hashtag-Count Pairs** * **Totaling the Hashtag Counts So Far** * **Specifying the Method to Call for Every RDD** * **Starting the Spark Stream** | |  |  |
|  |  | **16.8 INTERNET OF THINGS AND DASHBOARDS**   * **This Section’s Examples**   **16.8.1 Publish and Subscribe**  **16.8.2 Visualizing a PubNub Sample Live Stream with a Freeboard**   * **Dashboard** * **Signing up for Freeboard.io** * **Creating a New Dashboard** * **Adding a Data Source** * **Adding a Pane for the Humidity Sensor** * **Adding a Gauge to the Humidity Pane** * **Adding a Sparkline to the Humidity Pane** * **Completing the Dashboard**   **16.8.3 Simulating an Internet-Connected Thermostat in Python**   * **Installing Dweepy** * **Invoking the simulator.py Script** * **Sending Dweets**   **16.8.4 Creating the Dashboard with Freeboard.io**  **16.8.5 Creating a Python PubNub Subscriber**   * **Message Format** * **Importing the Libraries** * **List and DataFrame Used for Storing Company Names and Prices** * **Class SensorSubscriberCallback** * **Function Update** * **Configuring the Figure** * **Configuring the FuncAnimation and Displaying the Window** * **Configuring the PubNub Client** * **Subscribing to the Channel** * **Ensuring the Figure Remains on the Screen** | |  |  |
|  |  | **16.9 WRAP-UP** | |  |  |
| **CƏMİ:** | | | | **90** | **45M/45L** |
| **ƏDƏBİYYAT SİYAHISI** | | | |  |  |
| **1. References :**   1. [**http://www.deitel.com/books/cpphtp5/index.html**](http://www.deitel.com/books/cpphtp5/index.html)   **2. Following web sites may be referred:**   1. [**http://www.osborne.com**](http://www.osborne.com)   **3. The required material for practical can be downloaded from the site:**   1. [**http://www.sun.java.com**](http://www.sun.java.com) | | | |  |  |
| **Python for Programmers, First Edition**  **By: Paul J. Deitel; Paul Deitel; Harvey Deitel**  **Publisher: Prentice Hall**  **Pub. Date: April 12, 2019**  **Pages in Print Edition: 640** | | | |  |  |
| **The professional programmer’s Deitel® guide to Python® with introductory artificial intelligence case studies Written for programmers with a background in another high-level language, Python for Programmers uses hands-on instruction to teach today’s most compelling, leading-edge computing technologies and programming in Python–one of the world’s most popular and fastest-growing languages. Please read the Table of Contents diagram inside the front cover and the Preface for more details. In the context of 500+, real-world examples ranging from individual snippets to 40 large scripts and full implementation case studies, you’ll use the interactive IPython interpreter with code in Jupyter Notebooks to quickly master the latest Python coding idioms. After covering Python Chapters 1-5 and a few key parts of Chapters 6-7, you’ll be able to handle significant portions of the hands-on introductory AI case studies in Chapters 11-16, which are loaded with cool, powerful, contemporary examples. These include natural language processing, data mining Twitter® for sentiment analysis, cognitive computing with IBM® Watson™, supervised machine learning with classification and regression, unsupervised machine learning with clustering, computer vision through deep learning and convolutional neural networks, deep learning with recurrent neural networks, big data with Hadoop®, Spark™ and NoSQL databases, the Internet of Things and more. You’ll also work directly or indirectly with cloud-based services, including Twitter, Google Translate™, IBM Watson, Microsoft® Azure®, OpenMapQuest, PubNub and more. Features 500+ hands-on, real-world, live-code examples from snippets to case studies IPython + code in Jupyter® Notebooks Library-focused: Uses Python Standard Library and data science libraries to accomplish significant tasks with minimal code Rich Python coverage: Control statements, functions, strings, files, JSON serialization, CSV, exceptions Procedural, functional-style and object-oriented programming Collections: Lists, tuples, dictionaries, sets, NumPy arrays, pandas Series & DataFrames Static, dynamic and interactive visualizations Data experiences with real-world datasets and data sources Intro to Data Science sections: AI, basic stats, simulation, animation, random variables, data wrangling, regression AI, big data and cloud data science case studies: NLP, data mining Twitter®, IBM® Watson™, machine learning, deep learning, computer vision, Hadoop®, Spark™, NoSQL, IoT Open-source libraries: NumPy, pandas, Matplotlib, Seaborn, Folium, SciPy, NLTK, TextBlob, spaCy, Textatistic, Tweepy, scikit-learn®, Keras and more Accompanying code examples are available here: http://ptgmedia.pearsoncmg.com/imprint\_downloads/informit/bookreg/9780135224335/9780135224335\_examples.zip. Register your product for convenient access to downloads, updates, and/or corrections as they become available. See inside book for more information.** | | | | | |
| Part 1-Python Fundamentals  1 Introduction to Computers and Python  2 Introduction to Python Programming  3 Control Statements; Program Development  4 Functions  5 Lists and Tuples (This is the sample chapter enclosed with this proposal. We also enclose a   Part 2-Python Data Structures, Files and Databases  6 Arrays  7 Sets and Dictionaries  8 Strings: A Deeper Look  9 File and Exceptions  10 SQL Databases   Part 3-Python High-End Topics  11 Object-Based Programming: Classes and Objects  12 Object-Oriented Programming: Inheritance and Polymorphism  13 tkinter GUI  14 turtle Graphics and tkinter-Based Canvas Graphics  15 Concurrency and Parallelism  16 Game Programming with PyGame  17 Python Other Topics   Part 4-Python-Based Data-Science Case Studies  18 Natural Language Processing (NLP)  19 Data Mining Twitter: Web Services and JSON  20 Supervised Machine Learning  21 Unsupervised Machine Learning  22 Deep Learning  23 Reinforcement Learning  24 NoSQL and NewSQL Databases  25 Big Data with Hadoop  26 Big Data with Spark; Internet of Things (IoT)  27 Special Feature: IBM Watson Analytics and Cognitive Computing   Appendices  A Operator Precedence Chart  B ASCII Character Set  C Number Systems  D Testing and Debugging  E Integrated Development Environments (IDEs)  F Getting Your Questions Answered  G Suggested Capstone Projects and Thesis Topics | | | | | |
| Written for developers with a background in any high-level language, Introduction to Python and Data Science for Programmers explores the Python language and Python APIs in depth, applying the Deitels’ signature live-code approach to teaching programming. Paul Deitel and Dr. Harvey M. Deitel present concepts in the context of fully tested programs, complete with syntax shading, code highlighting, line-by-line code walkthroughs, and program outputs. They feature hundreds of complete Python programs with nearly 20,000 lines of proven Python code, and hundreds of tips to help you build robust applications. You’ll start with an introduction to Python using an early classes and objects approach, and then rapidly move on to more advanced topics.  Throughout, you’ll enjoy the Deitels' classic treatment of object-oriented programming. By the time you’re finished, you’ll have everything you need to build industrial-strength Python applications.  The DEITEL® Developer Series is designed for practicing programmers. The series presents focused treatments of modern technologies, including Python, Java™, C++, .NET, web services, Internet and web development, and more.  <https://www.twirpx.com/file/2797262/>  <https://newbooksreleases.com/category/new-books/computers>  <https://newbooksreleases.com/a-first-course-in-fuzzy-logic.html> | | | | | |

