5.1 Data wrangling

Learning goals

- Define data wrangling.
- List six steps of data wrangling process.
- Compare data wrangling with ETL.
- Use `pandas` to construct a dataframe.
- List `pandas` methods for working with dataframes.

What is data wrangling?

Data wrangling is the process of preparing source data for efficient and accurate analysis. Data-wrangling activities include removing missing values, formatting features uniformly, and appending related data from external sources. Data wrangling is sometimes called ***data munging*** or ***data preparation***.

Data wrangling has six steps.

Table 5.1.1: Steps of data wrangling.

Step	Description
Step 1: Discovering	Discovery, also called data exploration, familiarizes the data scientist with source data in preparation for subsequent steps.
Step 2: Structuring	Structuring data transforms features to uniform formats, units, and scales.
Step 3: Cleaning	Cleaning data removes or replaces missing and outlier data.
Step 4: Enriching	Enriching data derives new features from existing features and appends new data from external sources.
Step 5: Validating	Validating data verifies that the dataset is internally consistent and accurate.
Step 6: Publishing	Publishing data makes the dataset available to other data scientists by storing data in a database, uploading data to the cloud, or distributing data files.

Data exploration, structuring, cleaning, and enriching are discussed in separate sections of this material. Validating and publishing are not discussed further.

PARTICIPATION ACTIVITY 5.1.1: Data wrangling steps.

Original

	Country	Continent	SurfaceArea	Population	IndependenceDate	OfficialLa
0	Antarctica	Antarctica	13120000	0	NaT	
1	China	Asia	9572900	1277558000	1-Oct-49	M
2	Bangladesh	Asia	143998	129155000	1971	
3	Brazil	South America	8547403	170115000	1822-09-07	Por
4	India	Asia	3287263	1013662000	1947	
5	United States	North America	9363520	278357000	July 4, 1776	

Wrangled

	Country	Continent	SurfaceArea	Population	Density	IndependenceDate	C
0	China	Asia	9572900	1277558000	133	1949-10-01	
1	Bangladesh	Asia	143998	129155000	897	1971-05-24	
2	Brazil	South America	8547403	170115000	20	1822-09-07	
3	India	Asia	3287263	1013662000	308	1947-08-15	
4	Norway	Europe	385207	5379000	14	1905-05-17	

5	United States	North America	9363520	278357000	36	1776-07-04

Animation content:

Step 1: A dataset appears with columns Country, Continent, SurfaceArea, Population, IndependenceDate, Official Language, and Percentage. Percentage column is highlighted.
Step 2: Cells in the dataset with missing data are highlighted.
Step 3: Date formatting is standardized in the IndependenceDate column. Percents are converted to decimals in the Percentage column.
Step 4: The row containing Antarctica is removed from the dataset. The missing value in Percentage for China is replaced.
Step 5: A new row containing Norway is added to the dataset. A new column representing population Density is added.

Animation captions:

1. This dataset contains country information. Percentage is the fraction of population that speaks the official language.
2. None, NaN (Not a Number), and NaT (Not a Time) represent missing data.
3. Step 2, structuring data, standardizes IndependenceDate format and Percentage scale.
4. Step 3, cleaning data, removes Antarctica and replaces China's Percentage with the average value of other countries.
5. Step 4, enriching data, appends Norway and derives Density as (Population/SurfaceArea).

PARTICIPATION ACTIVITY 5.1.2: Data wrangling steps.

Map the data-wrangling step to the activity.

If unable to drag and drop, refresh the page.

Discovery **Cleaning** **Publishing** **Enriching** **Validating** **Structuring**

	Deriving a new feature, PerCapitaIncome, by dividing TotalIncome by Population.
	Changing the units of all distance features to meters.
	Uploading a dataset csv file to the cloud for analysis by other data scientists.
	Create a scatter plot to inspect raw data.
	Replacing missing values in a feature with the mean of known values.

Verifying that all two-letter state abbreviations in the State feature are valid states.

Reset

Extract, Transform, Load

Extract, Transform, Load (***ETL***) is a process that extracts data from transactional databases, transforms the data, and loads the data into an analytic database. ETL transforms data in a ***staging area***, such as a temporary database, prior to loading data to the analytic database. ***Extract, Load, Transform*** (***ELT***) is a variant of ETL that loads raw data directly to the analytic database and transforms the data in place.

ETL is similar to data wrangling. Both processes structure, clean, enrich, and publish data. However, data wrangling is usually an informal process, executed manually by data scientists on a static dataset. ETL is an automated process that repeatedly extracts new data from transactional databases. ETL is usually applied to larger data volumes and more sources than data wrangling.

ETL tools, also called ***data integration*** tools, extract and merge data from many different database systems. In principle, ETL tools can be used for data wrangling. However, because data wrangling is often manual and ad-hoc, data scientists usually prefer spreadsheets or programming languages such as Python, R, and SQL.

Table 5.1.2: Leading ETL tool vendors.

Vendor	Products	Comment
Informatica	Intelligent Cloud Services PowerCenter	Informatica is the leading data integration provider according to Gartner (2021). PowerCenter is Informatica's original stand-alone product. Intelligent Cloud Services is a newer cloud product.
IBM	Cloud Pak for Data Cloud Pak for Integration Data Replication	IBM is an older company with a long history of database products and services. Product capabilities are extensive but more complex than smaller vendors.
Microsoft	SQL Server Integration Services Azure Data Factory	SQL Server Integration Services is Microsoft's original stand-alone product. Azure Data Factory is a newer cloud product. Products initially focussed on SQL Server but now support a broad range of data sources.
Oracle	Data Integrator Big Data SQL Integration Cloud	Oracle is an older company with a long history of database products and services. Product capabilities are extensive but more complex than smaller vendors.
SAP	Data Intelligence Data Services Landscape Transformation	SAP products are relatively complex and expensive. Products are designed

	Replication Server Integration Suite	primarily for SAP data sources with limited integrations to non-SAP products.
Talend	Data Fabric Data Catalog	Founded in 2006, Talend is a relatively new ETL provider. The product line is less complex than larger, older competitors. Offers both open-source and commercial product versions.

Source: Gartner Magic Quadrant for Data Integration Tools (2021)

PARTICIPATION ACTIVITY 5.1.3: Data wrangling vs. ETL.

1) Which process structures, cleans, and enriches data?
   - ○ Data wrangling only
   - ○ ETL only
   - ○ Both data wrangling and ETL

2) Which process periodically extracts data from transactional database systems?
   - ○ Data wrangling only
   - ○ ETL only
   - ○ Both data wrangling and ETL

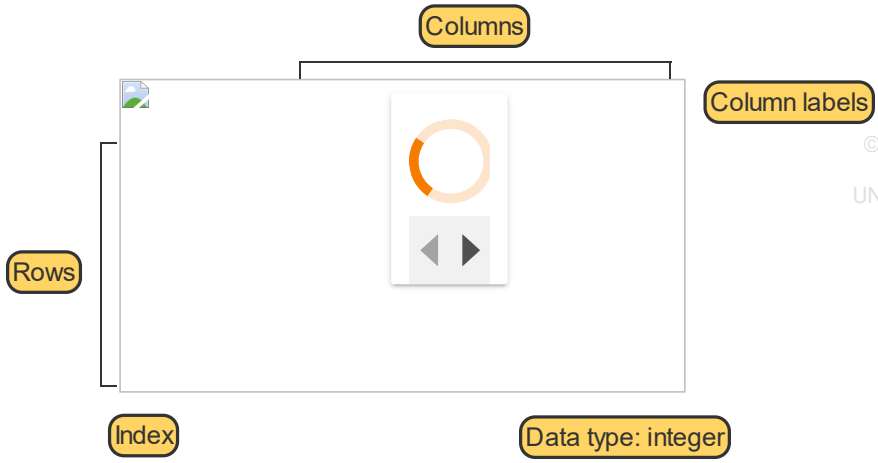
3) Which ETL tool vendors do not offer database system products?
   - ○ Informatica and Talend
   - ○ IBM and Microsoft
   - ○ Oracle and SAP

Data wrangling with Python

pandas is a Python package that supports data wrangling. ***DataFrame*** is a `pandas` class that stores and manipulates datasets. In this material, ***dataframe***, in lowercase, refers to a DataFrame object.

Dataframes consist of rows and columns, representing dataset instances and features. Each column has a data type. Rows and columns are identified by integer or string ***labels***. The set of row labels is called the ***index*** and the set of column labels is called ***columns***. Usually, row labels are automatically generated integers and column labels are manually specified strings.

PARTICIPATION ACTIVITY 5.1.4: Dataframes.



Animation content:

A dataframe with five rows and three columns. Row labels are 0 through 4. Column labels are Name, Continent, and Population. The index gives the row labels, 0 through 4. The data type of population is integer.

Animation captions:

1. The dataframe contains information about different countries.
2. Columns represent features. Column labels are usually strings.
3. Rows represent instances. Row labels, called the index, are usually automatically generated integers.
4. Each column has a data type. The type of the Population column is integer.

Selected `pandas` functions and methods that import, construct, display, and sort dataframes are described in the table below. The table includes all required parameters and important optional parameters, but the table excludes infrequently used optional parameters. For details on all methods and parameters, see pandas User Guide and pandas API Reference.

Table 5.1.3: Data wrangling with Python and pandas.

Method	Parameters	Description
`read_csv()`	`filepath_or_buffer` `sep=NoDefault.no_default`	Returns a dataframe constructed from a CSV file. `filepath_or_buffer` is a string containing the full path for the CSV file. When the file is in the sam directory as the code, only the file name is needed. `sep` specifies the charact

		that separates values in t CSV file.
`read_excel()`	`io` `sheet_name=0`	Returns a dataframe constructed from an Exce spreadsheet. `io` is a strin containing the full path fo the Excel file. When the fil is in the same directory a the code, only the file nan is needed. `sheet_name` a string or integer that specifies which Excel she to read.

`read_sql_table()`	`table_name`
	`con`
	`schema=None`
	`columns=none`
		Returns a dataframe constructed from an SQL table. `table_name` specifies the table name. `con` specifies a database server connection string. `schema` specifies the schema in the database server. `columns` specifie which table columns to include in the dataframe.

`DataFrame()`	`data=None`
	`index=None`
	`columns=None`
		Returns a new dataframe `data` specifies dataframe values as an array, dictionary, or another dataframe. `index` and `columns` specify row an column labels. The defaults `index=None` a `columns=None` generat integer labels.

`dataframe.at[]`	`index` `column`	Returns the `dataframe` value stored at `index` a `column`.
`dataframe.info()`	`verbose=None`	Returns information abou `dataframe`, such as number of rows and columns, data types, and memory usage. If `verbose=False`, shows only summary dataframe information and hides column details.
`dataframe.loc[]`	`indexRange` `columnRange`	Returns a slice of `dataframe`. `indexRange` specifies

		rows in the slice, as `startIndex:endInde` `columnRange` specifies columns in the slice as `startLabel:endLabe`

`dataframe.sort_values()`	`by`
	`axis=0`
	`ascending=True`
	`inplace=False`
		Sorts `dataframe` columns or rows. `by` specifies indexes or labels on which to sort. `axis` specifies whether to sort rows (0) or columns (1). `ascending` specifies whether to sort ascending or descending. `inplace` specifies whether to sort `dataframe` or return a new dataframe.

Data wrangling with Python.

Full screen

The Python code below constructs the dataframe `example`. The code then displays the contents of `example`, displays dataframe information, selects a slice, and sorts `example` on the `Continent` column.

Modify the code as follows:

Click the double right arrow icon to restart the kernel and run all cells.

- In the statement `example = pd.DataFrame()`, add a fourth column `'OfficialLanguage'` with the values in the animation above.
- Run the code again and verify your changes are correct.

Now, modify the code to create a new dataframe `country`:

- Import the file country.csv with the statement `country = pd.read_csv("country.csv")`.
- Display `country` and verify the data is identical to the animation above.
- Display dataframe information and review the data types of each column.
- Display the population of Norway using `country.at[]`.
- Sort `country` on the `Continent` column, ascending, and display the result.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

PARTICIPATION ACTIVITY 5.1.5: Data wrangling with Python and pandas.

1) Which `pandas` method imports a dataframe from a CSV file?

- `read_csv()`
- `read_excel()`
- `read_sql_table()`

2) Which `pandas` method returns information about a dataframe?

- `dataframe.at[]`
- `dataframe.info()`
- `dataframe.loc[]`

5.2 Manipulating data

Learning goals

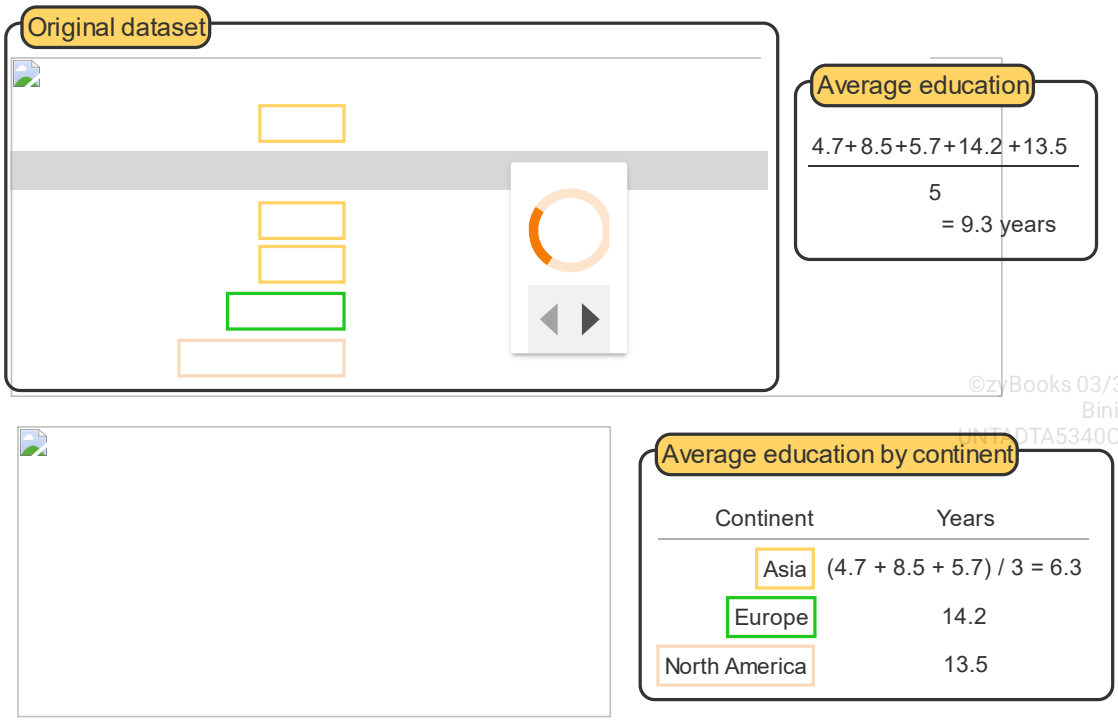
- Define data manipulation.
- Use group, calculate, and combine framework to compute descriptive statistics.
- Define and interpret frequency tables, contingency tables, and pivot tables.
- Compare groups in a dataset based on descriptive statistics.
- Compute descriptive statistics and tables using `pandas`.

Data manipulation

The first step of the data wrangling process is data discovery, or exploring patterns and trends within a dataset. Data exploration can be done visually through plots or figures, or numerically by comparing descriptive statistics.

Data manipulation is the process of organizing or subsetting a dataset to explore a research problem. Data manipulation is used to split datasets into multiple groups based on a categorical feature, or compare values of a dataset according to a specific condition. After data manipulation, descriptive statistics like the mean, median, or proportion can be calculated and compared across groups or conditions.

PARTICIPATION ACTIVITY 5.2.1: Manipulating data.



Animation content:

Step 1: A dataset appears with four features: Country, Continent, GDP, and EducationYears. The six countries in the rows of the dataset are Bangladesh, Brazil, China, India, Norway, and the United States.
Step 2: The second row in the dataset contains Brazil and is covered with a partially opaque box.
Step 3: A box appears containing the average education years for the five remaining countries.
Step 4: A world map appears with countries shaded by continent. Each country is highlighted in the dataset with the corresponding color for each continent.
Step 5: The mean education years for the three Asian countries (Bangladesh, China, and India) is 6.3. The mean education years for Europe (Norway) is 14.2 and for North America (United States) is 13.5.

Animation captions:

1. The United Nations has several programs designed to increase access to education. One measure of educational access is the average years in school.
2. GDP and EducationYears are unavailable for Brazil. Since data is missing, Brazil is filtered, or ignored, during data manipulation.
3. The mean, or average, EducationYears for the five countries in the dataset with data available is (4.7 + 8.5 + 5.7 + 14.2 + 13.5) / 5 = 9.3 years.
4. However, differences exist between countries, which affect access to education. Countries in the same continent or region are more likely to have similar education structures.
5. Grouping by continent highlights regional differences in education. Based on this data, students in Asia tend to spend less time in school than students in Europe and North America.

PARTICIPATION ACTIVITY 5.2.2: Data manipulation.

Refer to the following dataset.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters
Audi	A4	All	Gas	4	2
BMW	328Ci	Rear	Gas	6	3.6
Tesla	Model 3	All	Electric	NaN	NaN
Chevrolet	Malibu	Front	Gas	6	3.6
Ford	Mustang	Rear	Gas	8	5
Rolls-Royce	Ghost	Rear	Gas	12	6.6

1) Why is data manipulation an important part of data wrangling?

- Data manipulation allows comparison of multiple groups.
- Data manipulation results in a larger dataset.
- Data manipulation creates new features in a dataset.

2) Which research question requires data manipulation to answer?

- Is there a relationship between a car's gas mileage and engine size?
- Is there a relationship between a car's gas mileage and drive type?
- Is there a relationship between a car's gas mileage and number of cylinders?

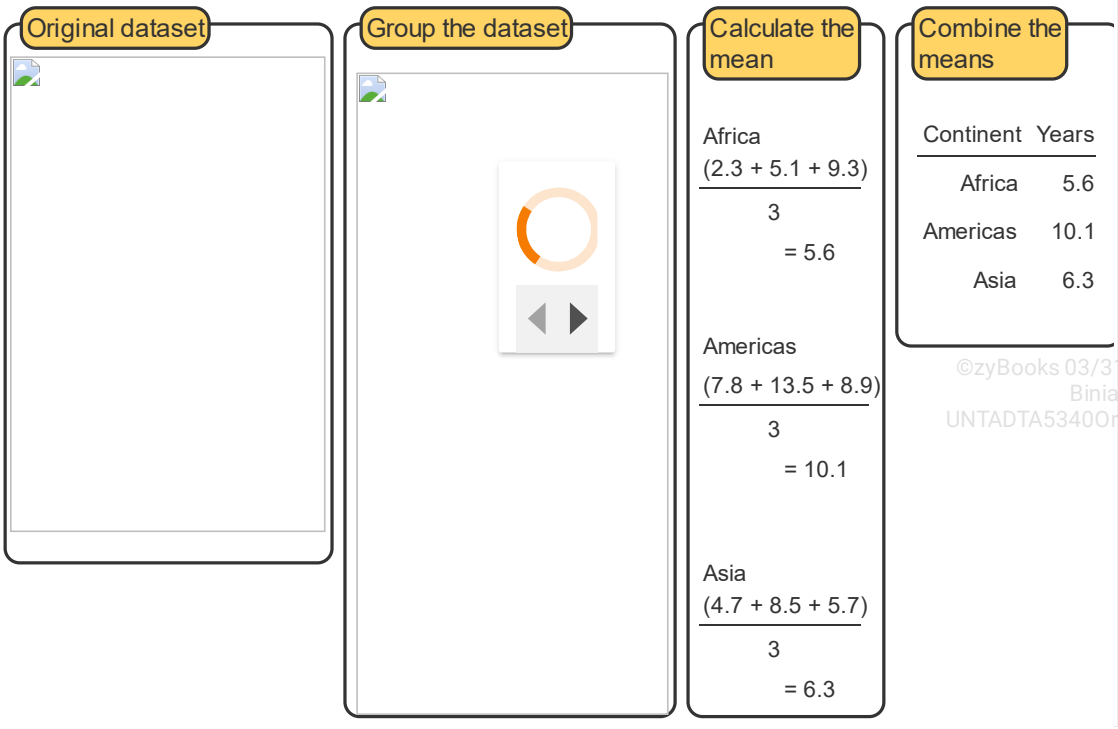
3) Which feature could be used to group the dataset?

- Manufacturer
- Drive
- Liters

Grouping data

Grouping is used to separate a dataframe into subsets based on levels of a categorical feature. In some cases, a different analysis or model may be applied to each group. In other cases, grouping may be a temporary operation for calculating group sizes or descriptive statistics like group means. A ***frequency table*** is a table containing group sizes for a categorical feature.

PARTICIPATION ACTIVITY 5.2.3: Data summaries using group, calculate, combine.



Animation content:

Step 1: A dataset appears containing three columns: Country, Continent, and Years.
Step 2: The dataset is separated into three smaller datasets: one containing all countries in Africa, one containing all countries in the Americas, and one containing all countries in Asia.
Step 3: The mean number of years in school is calculated for the three smaller datasets.
Step 4: Means move into a single table.

Animation captions:

1. A data scientist would like to compare the mean years of schooling (Years) for each continent.
2. The dataset is grouped into three subsets: one subset for Africa, one subset for the Americas, and one subset Asia.
3. After grouping, the mean is calculated for each subset.
4. Means for each continent are combined into a single table. Based on the table, countries in Africa have the lowest mean years in school, then Asia, then the Americas.

PARTICIPATION ACTIVITY 5.2.4: Grouping data.

Refer to the following dataset. Suppose Drive is used as a grouping feature.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters
Audi	A4	All	Gas	4	2
BMW	328Ci	Rear	Gas	6	3.6
Tesla	Model 3	All	Electric	NaN	NaN
Chevrolet	Malibu	Front	Gas	6	3.6
Ford	Mustang	Rear	Gas	8	5
Rolls-Royce	Ghost	Rear	Gas	12	6.6

1) Calculate the group size for rear-wheel drive cars.

- 1
- 2
- 3

2) A table containing group size for rear-wheel drive cars is a _____ table.

- frequency
- size
- count

3) Calculate the mean number of liters for rear-wheel drive cars.

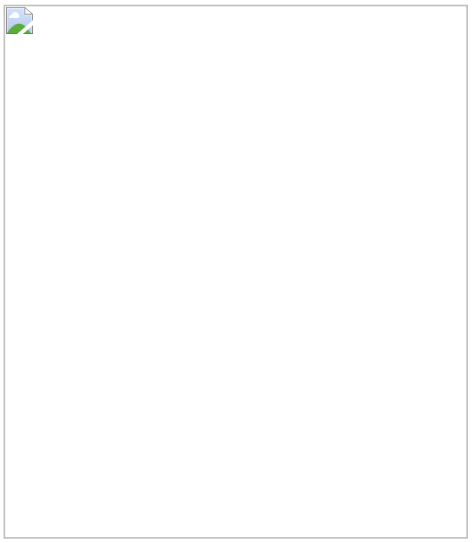
- 4.2
- 5.1
- NaN

Pivot tables

A ***pivot table*** calculates and displays descriptive statistics after grouping based on values of two categorical features. One categorical feature is assigned to the pivot table's rows, and another categorical feature is assigned to the columns. A ***contingency table*** is a special case of a pivot table in which the descriptive statistic is the number of instances in each combination of categorical features.

PARTICIPATION ACTIVITY 5.2.5: Pivot tables.

Original dataset



Pivot table with number of instances

	Internet access			
Continent	Low	Moderate	High	Very high
Africa	36	8	1	0
Americas	7	10	8	1
Asia	13	9	10	8
Europe	0	3	26	7
Oceania	1	1	2	0

Pivot table with mean years in school

	Internet access			
Continent	Low	Moderate	High	Very high
Africa	4.8	7.5	9.5	--
Americas	7.3	9.2	9.7	13.3
Asia	6.3	9.6	10.5	10.1
Europe	--	11.0	11.6	12.8
Oceania	7.9	10.8	12.7	--

Animation content:

Step 1: A dataset with five features: Country, Continent, Years, and internet access.
Step 2: The column containing Continent is highlighted. Continent has five unique values: Africa, Americas, Asia, Europe, and Oceania.
Step 3: The column containing internet access is highlighted. Internet access has four unique values: Low, Moderate, High, Very high.
Step 4: A pivot table with five rows (Africa, Americas, Asia, Europe, and Oceania) and four columns

(Low, Moderate, High, Very high) appears. Values in the table describe the number of instances in each group.
Step 5: A second pivot table with the same row/column structure appears. Values in the table describe the mean years in school in each group.

Animation captions:

1. Pivot tables provide the number of instances that share a combination of two categorical features.
2. The row feature's unique values are listed on the pivot table's rows.
3. The column feature's unique values are listed on the pivot table's columns.
4. A descriptive statistic like the number of instances is added to each row/column combination. Ex: 26 European countries have high internet access. 13 Asian countries have low internet access.
5. Pivot tables may contain descriptive statistics for additional features. Ex: The mean years in school for European countries with high internet access is 11.6 years, and the mean for Asian countries with low internet access is 6.3 years.

Pivot table.

Data analysis software such as Tableau or Excel often include interfaces for creating pivot tables. Ex: To re-create the pivot tables from the previous animation:

- Click the "Columns" button to the right of the dataframe. Click the button to enable "Pivot Mode".
- Click and drag "Continent" to the "Row Groups" section.
- Click and drag "Internet access" to the "Column Labels" section.
- Click and drag "Years" in school to the "Values" section. By default, the sum of all values in each group is calculated. Click "sum(Years)" and change this to "count(Years)" or "avg(Years)".

PARTICIPATION ACTIVITY 5.2.6: Pivot table.

Consider the cars dataset.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters
Audi	A4	All	Gas	4	2
BMW	328Ci	Rear	Gas	6	3.6
Tesla	Model 3	All	Electric	NaN	NaN
Chevrolet	Malibu	Front	Gas	6	3.6
Ford	Mustang	Rear	Gas	8	5
Rolls-Royce	Ghost	Rear	Gas	12	6.6

The pivot table below was created from the cars dataset.

	All	Front	Rear
Electric	1	0	0
Gas	1	1	3

1) Which is the row feature?

- Drive
- EngineType
- Cylinders

2) Which is the column feature?

- Drive
- EngineType
- Cylinders

3) What are the values in the pivot table?

- Group sizes
- Group means for cylinder
- Group medians for cylinder

Data manipulation in Python

The `pandas` package contains two methods for data manipulation. `df.groupby()` splits a dataframe `df` into subsets, and `df.pivot_table()` creates pivot tables based on two categorical features. Both `df.groupby()` and `df.pivot_table()` can be combined with `pandas` descriptive statistics methods.

`df.groupby()` sets the grouping feature using the `by` parameter. Missing values for the grouping feature can be removed by setting `dropna=True` or grouped into a separate category by setting `dropna=False`. Additional parameters for `df.groupby()` can be found in the group by documentation.

`df.pivot_table()` takes several parameters. The `value` specifies the values in the pivot table's elements. The feature in the pivot table's rows is specified using `index` and the feature in the pivot table's columns is `columns`. `aggfunc` specifies a function to apply to the values in each row/column combination within the pivot table. The default aggregate function is `np.mean`.

Manipulating datasets with Python.

Full screen

The Python code below imports the country dataset and explores the data using `pandas` functions and methods.

- Click the double right arrow icon to restart the kernel and run all cells.
- Examine the code below.
- Modify the code to explore different features and descriptive statistics.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

PARTICIPATION ACTIVITY 5.2.7: Data manipulation in Python.

1) Write the code that calculates the median for numerical features in the `cars` dataset based on the drive type (`Drive`).

Check Show answer

2) Write the code that creates a pivot table containing the median miles per gallon (`MPG`) for the `cars` dataset with `Drive` on the rows and `EngineType` on the columns.

Check Show answer

3) Write the code that creates a contingency table for the `cars` dataset with `Drive` on the rows and `EngineType` on the columns.

Check Show answer

CHALLENGE ACTIVITY 5.2.1: Manipulating data.

537150.4174434.qx3zqy7

CHALLENGE ACTIVITY 5.2.2: Manipulating data using pandas.

537150.4174434.qx3zqy7

Start

This dataset contains information on credit card customers, such as the customer's average credit limit, number of credit ca number of physical visits to the bank, and number of visits to the bank's website.

- Use a `pandas` method to create a frequency table for `Total_visits_online`.
- Assign the table to `freqTable`.

The code provided contains all imports, loads the dataset, and prints the table.

main.py credit_card.csv

```
# Import packages and functions
import pandas as pd

# Load the dataset
df = pd.read_csv('credit_card.csv')

# Create a frequency table for Total_visits_online
# Your code goes here

# Display frequency table
print(freqTable)
```

1 2

Check Next level

5.3 Structuring data

Learning goals

- Transform features in a dataset to uniform formats.
- Define and compute standardized features.
- Define and compute normalized features.
- Identify and transform overloaded features.
- Use `pandas` and `scikit-learn` methods to structure data.

Formatting data

All data within a single column and similar data in multiple columns should be stored in a uniform format. Ex:

- All dates and times might be stored as the datetime data type, with a 24-hour clock in Coordinated Universal Time (UTC).
- All lengths might be stored as meters.
- All percentages might be stored as decimal values between 0 and 1, rather than integers between 0 and 100.
- All names of people might be stored as 'FirstName LastName' with no prefix, suffix, or middle initial.

A uniform storage format facilitates aggregating and comparing data within and across columns. Standardizing the storage format also minimizes analysis errors.

Although storage formats should be uniform, display formats may vary according to the audience. Ex: In the animation below, Gross Domestic Product (GDP) is stored as integer dollars but might be displayed as billions of dollars.

PARTICIPATION ACTIVITY 5.3.1: Formatting data.

Original

	Country	Continent	SurfaceArea	GDP	IndependenceYear	OfficialL
0	China	asia	9572900	13.18T	1-Oct-49	
1	Bangladesh	Asia	143998	350B	1971	
2	Brazil	South America	8547403	1.87T	1822-09-07	Pc
3	India	Asia	3287263	2.72T	1947	
4	United States	North America	9363520	20.65T	July 4, 1776	

Formatted

	Country	Continent	SurfaceArea	GDP	IndependenceDate	Offic
0	China	Asia	9572900	13180000000000	1949-10-01	
1	Bangladesh	Asia	143998	350000000000	1971-05-24	
2	Brazil	South America	8547403	1870000000000	1822-09-07	
3	India	Asia	3287263	2720000000000	1947-08-15	
4	United States	North America	9363520	20650000000000	1776-07-04	

Animation content:

A dataset appears. Modifications are made to the dataset as described in the captions.

Animation captions:

1. Continent names are formatted with a leading capital letter.
2. SurfaceArea values are in square kilometers. No conversion is necessary.
3. Gross Domestic Product abbreviations T (trillions) and B (billions) are converted to integer dollars.
4. IndependenceYear is renamed IndependenceDate. Dates are stored as a date data type.
5. Percentage is the fraction of the population that speaks the official language. Numbers are converted to decimal values between 0 and 1.

PARTICIPATION ACTIVITY 5.3.2: Formatting data.

Refer to the following dataset. Match the column to the formatting inconsistency.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters	MPG
Audi	A4	All	Gas	4	2000.0	24
BMW	328 Ci	Rear	Gas	6	3.6	20
Chevrolet	Malibu	Front	Gas	6	3.6	18
FORD	Mustang	RWD	Gas	6	3.7	19
Rolls-Royce	Ghost	Rear	Gas	12	6.6	12
Subaru	Outback	4 wheel	Gas	4	2.5	22

If unable to drag and drop, refresh the page.

Liters **Drive** **Model** **Manufacturer**

	No consistency issues
	Inconsistent text format
	Inconsistent units
	Inconsistent categorical values

Reset

Feature scaling

The numeric features in a dataset often have different scales. In some datasets, scales may differ by orders of magnitude. Many algorithms execute faster or generate better results when scales are similar or identical. ***Feature scaling*** converts numeric features to uniform ranges. Two common feature scaling methods are standardization and normalization.

Standardization converts features to a range centered at 0, with 1 representing a standard deviation: $$x_{standardized} = \dfrac{x_{original} - \mu_x}{\sigma_x}$$ \(\mu_x\) is the mean and \(\sigma_x\) is the standard deviation of feature \(x\). The standardized value is called a ***z-score***. Since each unit represents one standard deviation, most z-scores fall between -2 and 2.

Normalization converts features to the range [0,1]: $$x_{normalized} = \dfrac{x_{original} - min_x}{max_x - min_x}$$

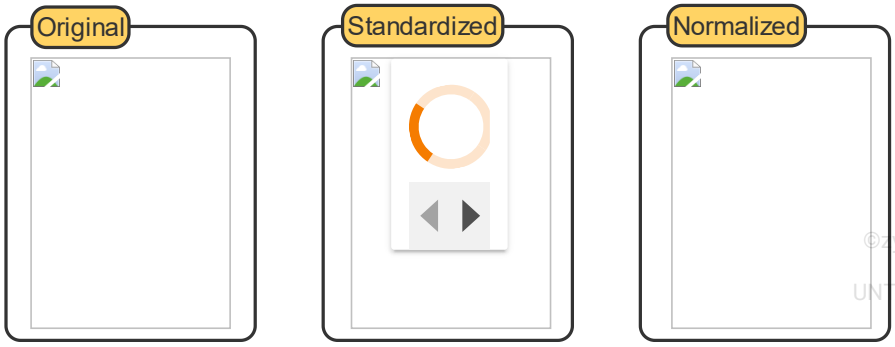
Standardization is usually preferred over normalization, since standardization positions values relative to the mean and standard deviation. Normalization is useful when algorithms require all features on identical scales.

Standardization is best when outliers are present. Standardized values are not skewed by outliers, but most normalized values are compressed into a small range.

Terminology

Feature scaling terminology varies. Standardization is sometimes called z-score normalization. Normalization is sometimes called min-max scaling.

PARTICIPATION ACTIVITY 5.3.3: Feature scaling.



Animation content:

Step 1: A dataset appears with two columns: Price and Age. Price ranges from 90300 to 269500. Age ranges from 14 to 28.

Step 2: Standardized values of Price are calculated using the formula (Price-170590)/74010. Standardized price ranges from -1.08 to 1.34.
Step 3: Standardized values of Age are calculated using the formula (Age-21.2)/5.8. Standardized age ranges from -1.23 to 1.16.
Step 4: Normalized values of Price are calculated using the formula (Price-90300)/(269500-90300). Normalized price ranges from 0 to 1.
Step 5: Normalized values of Age are calculated using the formula (Age-14)/(28-14). Normalized age ranges from 0 to 1.

Animation captions:

1. The housing dataset has the features Price and Age. The feature scales differ by orders of magnitude.
2. Standardized Price values are computed from the mean, 170,590, and standard deviation, 74,010.
3. Standardized Age values are computed from the mean, 21.2, and standard deviation, 5.8.
4. Normalized Price values are computed from the minimum, 90,300, and maximum, 269,500.
5. Normalized Age values are computed from the minimum, 14, and maximum, 28.

PARTICIPATION ACTIVITY 5.3.4: Standardization.

Refer to the datasets in the animation above.

1) The house with a price of 269,500 is 1.34 standard deviations above the mean.
   - ○ True
   - ○ False

2) The house with a price of 150,500 is 0.27 standard deviations above the mean.
   - ○ True
   - ○ False

3) The house with a price of 244,650 is 0.86 standard deviations above the mean.
   - ○ True
   - ○ False

4) The standardized scales for Price and Age are identical.
   - ○ True
   - ○ False

©zyBooks 03/31/24 10:45 2087217
Biniam abebe
UNTADTA5340OrhanSpring8wk22024

5) The normalized scales for Price and Age are identical.
   - ○ True
   - ○ False

PARTICIPATION ACTIVITY 5.3.5: Normalization.

1) Assuming car weight has a minimum of 500 pounds and a maximum of 3,000 pounds, what is the normalized value of a car weighing 2,000 pounds?

Ex: 1.23

Check Show answer

2) Assuming engine size is a minimum of 1.2 liters and a maximum of 6.4 liters, what is the normalized value of 3.5 liters?

Ex: 1.23

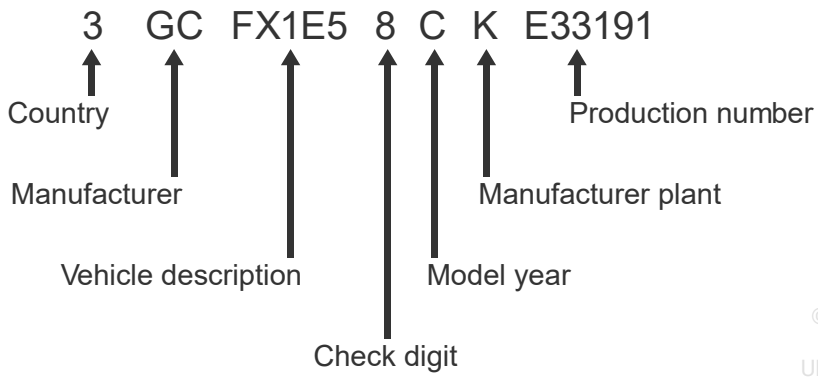
Check Show answer

Unpacking data

Some features encode multiple types of data as a single value. Ex: A PartCode feature may include characters that describe the part color. An ***overloaded feature*** encodes multiple types of data. A ***simple feature*** contains only one type of data.

To simplify analysis, data encoded in an overloaded feature should be extracted and stored as new, simple features. The overloaded feature usually remains in the dataset as reference information.

PARTICIPATION ACTIVITY 5.3.6: Vehicle Identification Number (VIN).



Animation content:

A vehicle identification number (VIN) appears. Parts of the VIN are labeled as described in the captions.
VIN: 3 GC FX1E5 8 C K E33191

Animation captions:

1. The first character of a VIN encodes the country of manufacture. "3" is the code for Mexico.
2. Characters 2 and 3 encode the manufacturer and, in some cases, division. "GC" encodes the General Motors Trucks division.
3. Characters 4 through 8 encode information about the vehicle, such as body, engine, and transmission types.
4. Character 9 is a "check digit" that is used to validate the VIN. An incorrect digit means the VIN is invalid.
5. The remaining characters encode the model year, plant of manufacture, and sequential production number for the plant.

PARTICIPATION ACTIVITY 5.3.7: Unpacking data.

Refer to the animation above.

1) If all information encoded in the VIN is unpacked, how many new features are added to the dataset?
   - One
   - Seven
   - More than seven

2) When the VIN is unpacked, is the original VIN feature retained in the dataset?
   - Yes
   - No

3) How should the model year be represented when unpacked as a new feature?
   - The original code
   - A year number derived from the code
   - A datetime value derived from the code

Structuring data with Python

The Python language, and the `pandas` and `sklearn` packages, have many data structuring methods. Selected methods and functions that format, scale, and unpack data are described in the tables below. The tables include all required parameters and important optional parameters, but exclude infrequently used optional parameters.

`string[start:end]` is not a method but is useful for unpacking string columns and therefore included in the table.

Methods that change data contain an optional `copy` parameter. If `copy` is True, changes are returned in a new dataframe or array. If `copy` is False, changes are made to the input dataframe or array.

Table 5.3.1: Python data structuring methods.

Method	Parameters	Description
`string[start:end]`	*none*	Returns the substring of `string` that begins at the index `start` and ends at the index `end - 1`.

`string.capitalize()`
`string.upper()`
`string.lower()`
`string.title()`
	none	Returns a copy of `string` with the initial character uppercase, all characters uppercase, all characters lowercase, or the initial character of all words uppercase.

`to_datetime()`	`arg`	Converts `arg` to datetime data type and returns the converted object. Data type of `arg` may be int, float, str, datetime, list, tuple, one-dimensional array, Series, or DataFrame.
`to_numeric()`	`arg`	Converts `arg` to numeric data type and returns the converted object. Data type of `arg` may be scalar, list, tuple, one-dimensional array, or Series.

Table 5.3.2: pandas data structuring methods.

Method	Parameters	Description
`df.astype()`	`dtype` `copy=True`	Converts the data type of all dataframe `df` columns to `dtype`. To alter individual columns, specify `dtype` as `{col: dtype, col:dtype, . . .}`.

`df.insert()`	`loc`
	`column`
	`value`
		Inserts a new column with label `column` at location `loc` in dataframe `df`. `value` is a Scalar, Series, or Array of values for the new column.

Table 5.3.3: sklearn data structuring methods.

Method	Parameters	

`preprocessing.scale()`	`X`
	`axis=0`
	`with_mean=True`
	`with_std=True`
	`copy=True`
		Sta
		inp
		Arr
		ax
		wh
		sta
		co

		(1) wi cer the wi sca tha sta

`preprocessing.MinMaxScaler().fit_transform()`	`feature_range=(0,1)`
	`copy=True`
	`X`
		No inp fi pa typ Da fe sp sca fe an Mi pa

Structuring data with Python.

Full screen

The Python code below standardizes and normalizes housing data.

- Click the double right arrow icon to restart the kernel and run all cells.
- Add statements that display the original, standardized, and normalized data.
- Verify that the original, standardized, and normalized data is the same as in the feature scaling animation above.

PARTICIPATION ACTIVITY 5.3.8: Feature scaling in Python.

The `carsMPG` dataset contains the miles per gallon, or MPG, of a sample of cars.

1) Complete the code to standardize miles per gallon.

```
preprocessing.
[ ]
(carsMPG)
```

Check Show answer

2) Complete the code to normalize miles per gallon.


```
preprocessing.
[ ]
(carsMPG)
```

Check Show answer

CHALLENGE ACTIVITY 5.3.1: Structuring data.

537150.4174434.qx3zqy7

CHALLENGE ACTIVITY 5.3.2: Structuring data using scikit-learn.

537150.4174434.qx3zqy7

Start

This dataset contains information on credit card customers, such as the customer's average credit limit, number of credit ca number of physical visits to the bank, and number of visits to the bank's website.

- Standardize the data for the features `Avg_Credit_Limit` and `Total_Credit_Cards`.

The code contains all imports, loads the dataset, subsets the data, and displays five lines of the standardized data.

main.py | credit_card.csv

```
# Import packages and functions
import pandas as pd
from sklearn import preprocessing

# Load the dataset
df = pd.read_csv('credit_card.csv')

# Create a new dataframe with two features
dfCredit = df[['Avg_Credit_Limit', 'Total_Credit_Cards']]

# Standardize dataframe and return as an array
standardizedArray = # Your code goes here

# Convert standardized array to dataframe
dfCreditStandardized = pd.DataFrame(standardizedArray, columns=['Avg_Credit_Limit', 'Total_Credit_Cards

# Print five lines of the standardized data
print(dfCreditStandardized[376:381])
```

1 2

Check Next level

5.4 Cleaning data

Learning goals

- List types of dirty data.
- Explain techniques to discard dirty data.
- Explain how to replace dirty data with imputed values.

- Use `pandas` methods to discard and impute data.

Dirty data

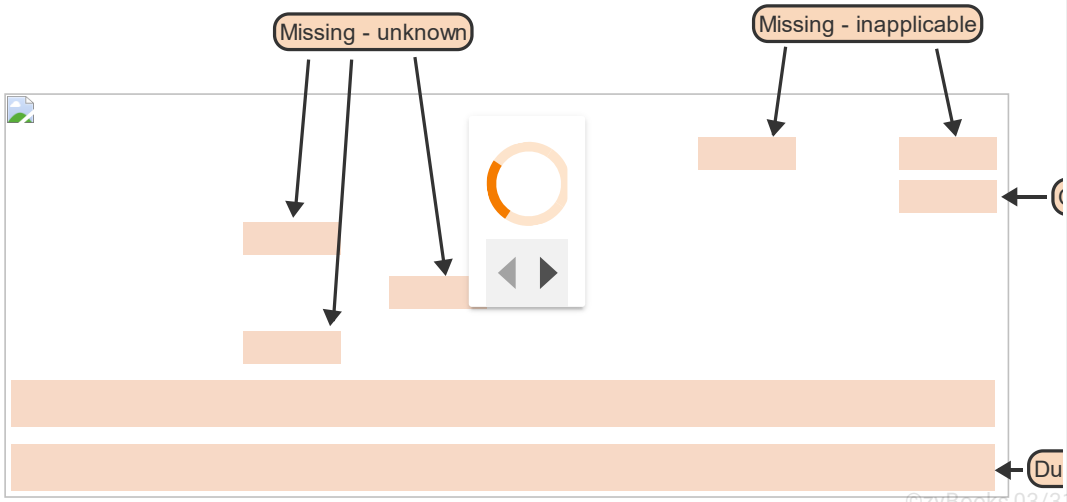
Raw datasets often contain missing, outlier, and duplicate data.

- ***Missing data*** is an unknown or inapplicable value. In a database, missing data is represented as `NULL`. In Python, missing data is represented as `NaN` (not a number), `NaT` (not a time), `None` (an unspecified object), or a blank value.
- ***Outlier data*** is a numeric value that is much larger or smaller than other values in the same feature. Outlier data is usually defined as two or three standard deviations from the feature mean.
- ***Duplicate data*** are two or more identical instances in a dataset. Duplicate instances are usually erroneous and should be removed.

Missing, outlier, and duplicate data are collectively called ***dirty data***. A ***dirty instance*** and a ***dirty feature*** contain dirty data.

Dirty data creates bias and inefficiencies in data analysis. Data scientists may struggle to interpret missing data. Values in erroneous duplicates appear too often and are weighted too heavily. Outliers skew results due to one potentially erroneous value. Consequently, missing, outlier, and duplicate data should be corrected or deleted.

PARTICIPATION ACTIVITY 5.4.1: Dirty data.



Animation content:

A dataset is displayed. Columns in the dataset are Country, Continent, SurfaceArea, Population, LifeExpectancy, and IndependenceYear.
Step 1: Antarctica has missing values for LifeExpectancy and IndependenceYear, which are highlighted.
Step 2: Bangladesh and Switzerland have missing values for Continent. Bolivia has a missing value for SurfaceArea.

Step 3: China's independence year is highlighted.
Step 4: The United States appears in the dataset twice.

Animation captions:

1. Antarctica has no permanent population and is not an independent country. So life expectancy and independence year are inapplicable.
2. Bangladesh, Bolivia, and Switzerland have continents and surface areas. So the missing values are unknown.
3. China's independence year is greater than three standard deviations from the mean and is thus an outlier.
4. The United States instances are duplicates.

PARTICIPATION ACTIVITY 5.4.2: Dirty data.

Refer to the following dataset.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters
Audi	A4	All	Gas	4	2
BMW	328Ci	Rear	Gas	6	3.6
Tesla	Model 3	All	Electric	NaN	NaN
Chevrolet	Malibu	Front	Gas	6	3.6
Ford	Mustang	Rear	Gas		5
Chevrolet	Malibu	Front	Gas	6	3.6
Rolls-Royce	Ghost	Rear	Gas	12	6.6

Match the term to the vehicle that illustrates the term.

If unable to drag and drop, refresh the page.

Duplicate **Missing - unknown** **Outlier** **Missing - inapplicable**

	Rolls-Royce Ghost
	Tesla Model 3
	Chevrolet Malibu
	Ford Mustang

Reset

Discarding data

Dirty data may be removed from a dataset by discarding instances, discarding features, or pairwise discarding.

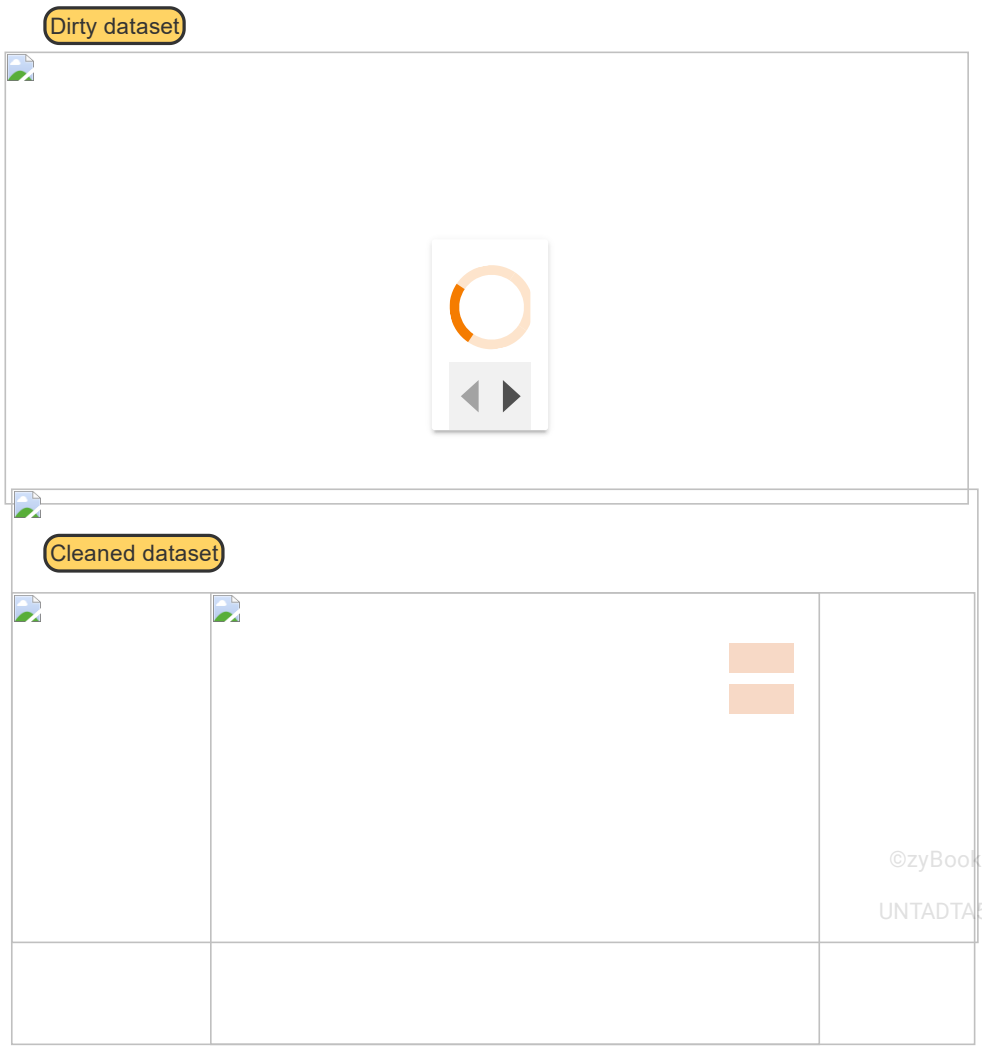
Discarding instances, also called ***listwise deletion*** or ***complete case removal***, removes dirty instances from the dataset. Dirty instances are usually discarded when:

- The dirty instances comprise a small percentage of the dataset.
- The dirty instances are random. When missing or outlier values are correlated with values in another feature, discarding dirty instances introduces bias.
- Instances are duplicates. Usually, duplicate instances are erroneous, and one instance should be discarded.

Discarding features removes dirty features that contain a high percentage of missing values, such as 60% or more. Discarding features does not usually apply to outlier data since, by definition, a small percentage of values can be outliers. Discarding features never applies to duplicate data.

Pairwise discarding retains dirty instances for some analyses and discards dirty instances for others. Instances are discarded only when an analysis uses a dirty feature. With pairwise discarding, the total number of instances varies for different analyses, which complicates comparisons and correlations. For this reason, pairwise discarding is not commonly used.

PARTICIPATION ACTIVITY 5.4.3: Discarding data.



Animation content:

The country dataset is displayed and modified as described in the captions.

Animation captions:

1. The duplicate United States instance is discarded.
2. The Saint Helena instance has several missing values and is discarded.
3. The LifeExpectancy feature has several missing and outlier values. The feature is discarded.
4. Independence year is missing for Antarctica and an outlier for China. These instances are pairwise discarded for analysis of independence year.

PARTICIPATION ACTIVITY 5.4.4: Discarding data.

Refer to the following dataset. Miles per gallon (mpg) is the distance a car travels on one gallon of gasoline. Miles per gallon electric (mpge) is the distance a car travels on 33.7 kWh of electricity.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters	mpg	mpge	Base price
Audi	A4	All	Gas	4	2	24	NaN	NaN
BMW	328Ci	Rear	Gas	6	3.6	20	NaN	NaN
Bentley	Continental	Rear	NULL	NaN	NaN	210	NaN	NaN
Tesla	Model 3	All	Electric	NaN	NaN	NaN	134	$35,990
Chevrolet	Malibu	Front	Gas	6	3.6	18	NaN	NaN
Ford	Mustang	Rear	Gas	6	3.7	NaN	NaN	NaN
Nissan	Leaf	Front	Electric	NaN	NaN	NaN	111	NaN
Rolls-Royce	Ghost	Rear	Gas	12	6.6	12	NaN	NaN

1) The Bentley Continental instance is a good candidate for discarding.
   - ○ True
   - ○ False

2) The base price feature is a good candidate for discarding.
   - ○ True
   - ○ False

3) The Tesla Model 3 instance is a good candidate for discarding.
   - ○ True
   - ○ False

4) The Tesla Model 3 and Nissan Leaf instances are good candidates for pairwise discarding.
   - ○ True
   - ○ False

©zyBooks 03/31/24 10:45 2087217
Biniam abebe
UNTADTA5340OrhanSpring8wk22024

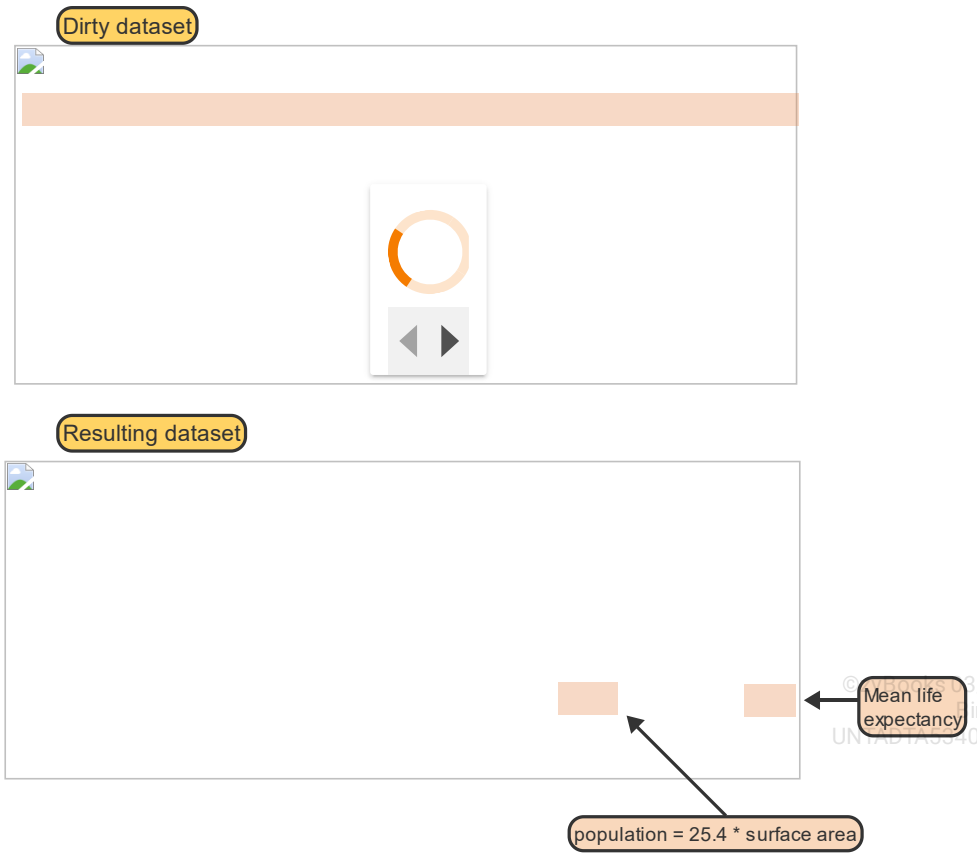
Imputing data

Imputing data replaces missing and outlier data with new values. Imputing is more complex than discarding but retains all instances and features. Data may be imputed in several ways:

- ***Hot-deck*** and ***cold-deck imputation*** replace missing and outlier data with a value from a randomly selected instance. In hot-deck imputation, the value is selected from other instances in the same dataset. In cold-deck imputation, the val is selected from a different dataset.
- ***Mean imputation*** replaces missing and outlier data with the mean value of the feature. Missing and outlier data are excluded from the computation of the mean.
- ***Regression imputation*** replaces missing and outlier data with a value computed from a regression model. In the regression model, the dependent variable is the dirty feature and the independent variables are other features. ***Stochastic regression imputation*** introduces uncertainty by adding or subtracting the regression variance to the new value. Regression models are discussed elsewhere in this material.

Regression imputation is valuable if the dirty feature is highly correlated with other features. If not, mean imputation is commonly used. Hot- and cold-deck imputation were common when less computer power was available to compute mean and regression but are not widely used today.

PARTICIPATION ACTIVITY 5.4.5: Imputing data.



Animation content:

The country dataset appears.
Step 1: The row containing Antarctica is removed from the dataset.
Step 3: The missing value for LifeExpectancy in Saint Helena is replaced with 68, which is the mean life expectancy in the dataset.
Step 4: The missing value for Population in Saint Helena is replaced with 7976, which is the result of applying the regression formula. Population = 25.4*SurfaceArea = 25.4*314 = 7976.

Animation captions:

1. Antarctica's population is an outlier and life expectancy is inapplicable. So Antarctica is discarded prior to imputing data.
2. Saint Helena's life expectancy is imputed from the mean of this dataset.
3. A linear regression on this dataset determines Population = 25.4 * SurfaceArea. So Saint Helena's population = 25.4 * 314 = 7976.

PARTICIPATION ACTIVITY 5.4.6: Imputing data.

Refer to the following dataset.

Manufacturer	Model	Drive	EngineType	Cylinders	Liters	MPG
Audi	A4	All	Gas	4	2	24
BMW	328 Ci	Rear	Gas	6	3.6	20
Bentley	Continental	Rear	Gas	NaN	NaN	210
Chevrolet	Malibu	Front	Gas	6	3.6	18
Ford	Mustang	Rear	Gas	6	3.7	NaN
Rolls-Royce	Ghost	Rear	Gas	12	6.6	12

1) Using mean imputation, what is the new mpg value for Ford Mustang?

- 18.5
- 47.3
- 56.8

2) Assume that a regression model determines `mpg = 40 - 2 * Cylinders - Liters`.
Using regression imputation, what is the new mpg value for Ford Mustang?

- 18.5
- 20
- 24.3

3) The mpg for the Bentley Continental is an outlier. Can a new value be imputed using the regression model `mpg = 40 - 2 * Cylinders - Liters`?

- Yes
- Only if the missing Cylinder and Liters values are first imputed.
- No

Cleaning data with Python

In Python, three special symbols represent missing values:

- `None` represents any missing Python object, such as a string.
- NaN represents a missing numeric value. NaN is a `NumPy` value, specified as `numpy.NaN`.
- NaT represents a missing datetime value. NaT is a `pandas` value, specified as `pandas.NaT`.

In addition, blank and 0 sometimes indicate a missing value, as in any tool.

The pandas DataFrame class has methods that identify dirty data and discard and impute values. Important data cleaning methods are described in the table below. The table includes all required parameters and important optional parameters but excludes infrequently used optional parameters.

Methods that change data contain an optional `inplace` parameter. If `inplace` is True, changes are made in the input dataframe. If `inplace` is False, changes are returned in a new dataframe.

Table 5.4.1: pandas data cleaning methods.

Method	Parameters	Description

`df.drop()`	`labels=None`
	`axis=0`
	`inplace=False`
		Removes rows (`axis=0`) or columns (`axis=1`) from dataframe `df`. `labels` specifies the labels of rows or columns to drop.

`df.drop_duplicates()`	`subset=None` `inplace=False`	Removes duplicate rows from `df`. `subset` specifies the labels of columns used to identify duplicates. If `subset=None`, all columns are used.
`df.dropna()`	`axis=0` `how='any'`	Removes rows (`axis=0`) or columns (`axis=1`)

	`subset=None` `inplace=False`	containing missing values from `df`. `subset` specifies labels on the opposite axis to consider for missing values. `how` indicates whether to drop the row or column if `any` or if `all` values are missing.
`df.duplicated()`	`subset=None`	Returns a Boolean series that identifies duplicate rows in `df`. `true` indicates a duplicate row. `subset` specifies the labels of columns used to identify duplicates. If `subset=None`, all columns are used.
`df.fillna()`	`value=None` `inplace=False`	Replaces NA and NaN values in `df` with `value`, which may be a scalar, dict, Series, or DataFrame.
`df.isnull()` `df.isna()`	*none*	Returns a dataframe of Boolean values. `True` in the returned dataframe indicates the corresponding value of the input `df` is None, NaT or NaN.

`df.mean()`	`axis=0`
	`skip_na=True`
	`numeric_only=None`
		Returns the mean values of rows (`axis=0`) or columns (`axis=1`) of `df`. `skipna` indicates whether to exclude unknown values in the calculation. `numeric_only` indicates whether to

		exclude non-numeric rows or columns.
`df.replace()`	`to_replace=None value=NoDefault.no_default inplace=False`	Replaces `to_replace` values in `df` with `value`. `to_replace` and `value` may be str, dict, list, regex, or other data types.

Cleaning data with Python.

Full screen

The Python code below imports and cleans automobile data. Make the following changes:

- At the end of each cell, add a statement that displays 'auto'.
- Add a new cell 5 that displays 'mean'.

Click the double right arrow icon to restart the kernel and run all cells.

- `auto.drop_duplicates()` eliminates the duplicate Chevrolet row.
- `auto.dropna()` eliminates the Bentley row with two unknown values.
- `auto.mean()` computes the mean values of Cylinders, Lliters, and mpg.
- `auto.fillna()` replaces the unknown Ford mpg value with the mean mpg value.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

PARTICIPATION ACTIVITY 5.4.7: Cleaning data with pandas.

1) Complete the code to replace missing values in the `cars` dataset with 20.

```
cars.
```

Check Show answer

2) Complete the code to remove the `MPG` feature from the cars dataset.

```
cars.
(axis=1, labels='MPG')
```

Check Show answer

3) Complete the code to remove duplicate rows from the `cars` dataset.

```
cars.
```

Check Show answer

CHALLENGE ACTIVITY 5.4.1: Cleaning data.

537150.4174434.qx3zqy7

CHALLENGE ACTIVITY 5.4.2: Cleaning data using pandas.

537150.4174434.qx3zqy7

Start

This dataset contains information on lifestyle measures such as the amount of sunshine, pollution, and happiness levels for major cities around the world.

- Drop instances with duplicate values of `outdoor_activities` from the dataset.

The code contains all imports, loads the dataset, and displays `rankingsClean.info()`.

main.py | rankings_example.csv

```
# Import packages and functions
import pandas as pd

# Load the dataset
rankings = pd.read_csv('rankings_example.csv')

# Remove instances with duplicate values of outdoor_activities
rankingsClean = # Your code goes here

rankingsClean.info()
```

1 2 3

Check Next level

5.5 Enriching data

Learning goals

- Identify sources of public data.
- Explain how to append features and instances to a dataset.
- List techniques for deriving new categorical and numeric features.
- Use `pandas` methods to append and derive data.

Appending data

Datasets can be enriched by appending new instances or features from external datasets. Leading sources of public datasets are described in the table below.

Table 5.5.1: Leading public datasets.

Name	Link	Description
Kaggle	kaggle.com	Over 50,000 datasets on a broad range of subjects. Also provides Jupyter notebooks that analyze the datasets.
FiveThirtyEight	data.fivethirtyeight.com	Datasets on politics, sports, science, economics, health, and culture, initially developed to support FiveThirtyEight publications.
University of California Irvine Machine Learning Repository	archive.ics.uci.edu	622 datasets, primarily in science, engineering, and business.
Data.gov	data.gov	U.S. government datasets on agriculture, climate, energy, maritime, oceans, and health.
World Bank Open Data	data.worldbank.org	Global datasets on subjects such as health, education, agriculture, and economics.
Nasdaq Data Link	data.nasdaq.com	Financial and economic datasets.

To append instances or features, prepare a subset of external data as follows:

1. Identify the external dataset of interest.
2. Identify a matching feature in the external and original datasets. The matching feature must uniquely identify instances of both datasets.
3. Usually, only a subset of the external dataset is of interest. Extract the subset, including the matching feature.
4. Structure and clean the subset, as described elsewhere in this material.

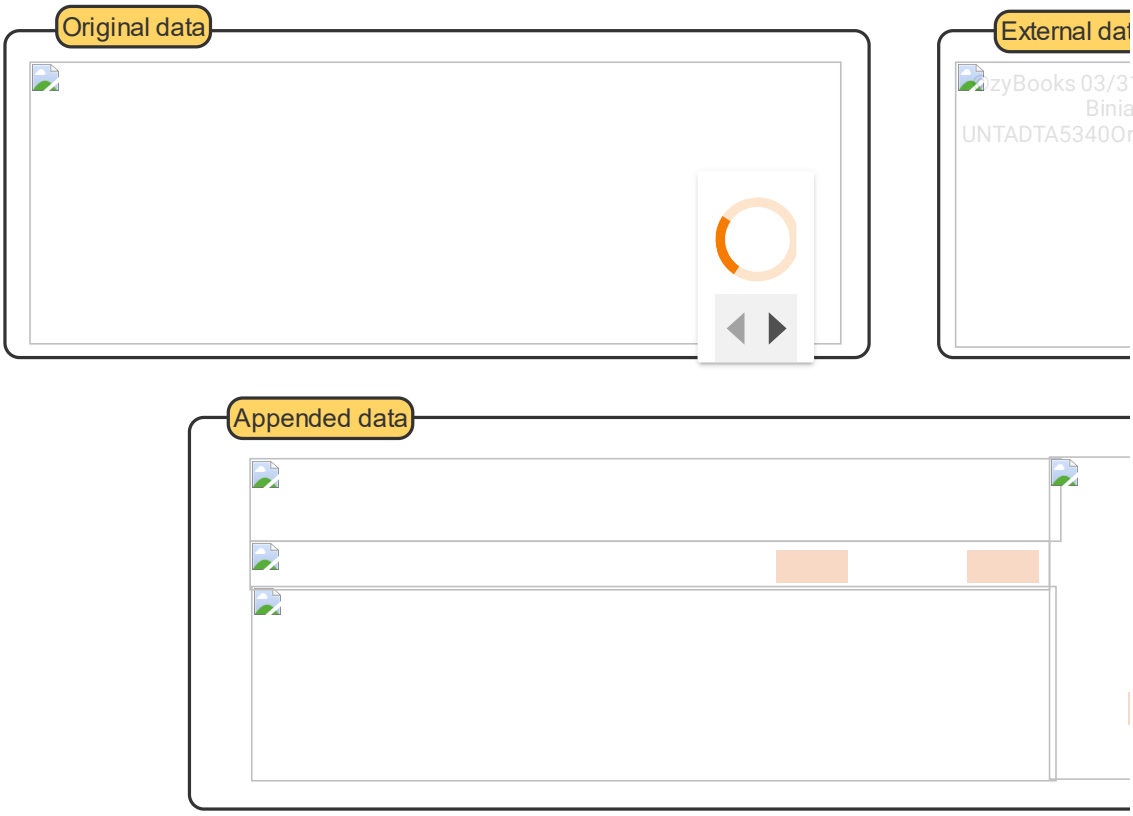
To append instances, insert subset instances to the original dataset. To append features, merge subset instances with original instances using the matching feature, as illustrated in the animation below.

Appending data may create missing data:

- When appending instances, missing data is created if the two datasets have different features.
- When appending features, missing data is created if instances of the two datasets do not match.

Discard or impute the new missing data, as described elsewhere in this material.

PARTICIPATION ACTIVITY 5.5.1: Appending data.



Animation content:

Step 1: A dataset containing the columns Country, Continent, GDP, and EducationYears appears. Countries in the first dataset are Bangladesh, China, India, Norway, and the United States.
Step 2: A second dataset appears containing the columns Name, Continent, and Population. Countries in the second dataset are Bangladesh, Brazil, China, India, and the United States.
Step 3: Country is highlighted in the first dataset and Name is highlighted in the second dataset.
Step 4: The two datasets are combined. A row for Brazil is added, with missing values for GDP and EducationYear.
Step 5: A column for Population appears in the new dataset. Norway was not represented in the second dataset, and so does not have a value for Population.
Step 6: Missing values are highlighted in the final dataset.

Animation captions:

1. Original dataset has features for country name, continent name, gross domestic product, and average years of education.
2. External data has a new Population feature and Brazil instance.
3. The matching feature is Country in the original dataset and Name in the external dataset.
4. Append Brazil instance. Since GDP and EducationYears features are not in the external dataset, Brazil values are NaN.
5. Append Population feature. Since Norway instance is not in external dataset, Population value is NaN.

6. Discard or impute new missing values, as described elsewhere in this material.

PARTICIPATION ACTIVITY 5.5.2: Appending data.

The following questions refer to the new dataset after the external dataset is appended to the original dataset.

Original dataset:

Manufacturer	Model	Drive	EngineLiters
Audi	A4	All	2.0
BMW	328 Ci	Rear	3.6
Chevrolet	Malibu	Front	3.6
Ford	Mustang	Rear	5.0

External dataset:

Manufacturer	Model	Drive	Cylinders
Audi	A4	All	4
BMW	328 Ci	Rear	6
Chevrolet	Malibu	Front	6
Rolls-Royce	Ghost	Rear	12

1) Which instance has a missing value in the Cylinders feature?

- No instances
- Ford Mustang
- Audi A4

2) Which instance has a missing value in the Drive feature?

- No instances
- Ford Mustang
- Rolls-Royce Ghost

3) Which feature has a missing value in the Rolls-Royce Ghost instance?

- No feature
- Drive
- EngineLiters

©zyBooks 03/31/24 10:45 2087217
Biniam abebe
UNTADTA5340OrhanSpring8wk22024

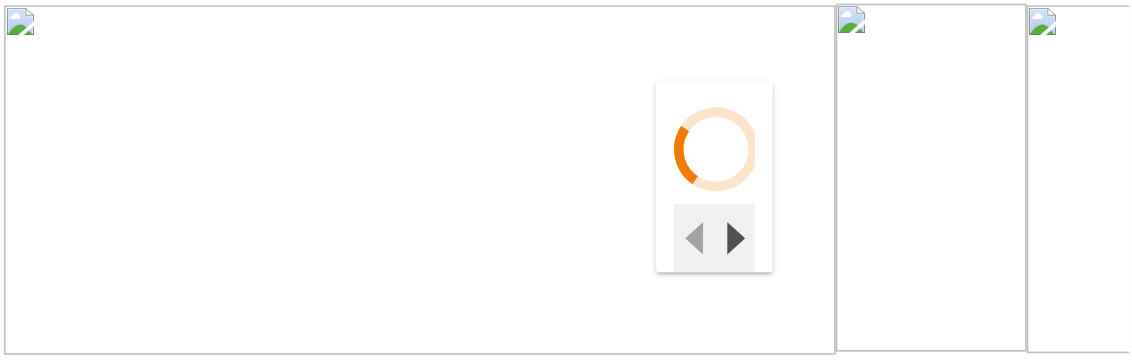
Deriving data

A new feature can be derived from existing features in several ways:

- Calculate a new feature from one or more existing features. Ex: Calculate a new feature as the logarithm of an existing feature. Ex: Calculate a new feature as the ratio of one existing feature to another.
- Convert an existing categorical feature to a new numeric feature, or vice versa. Ex: Convert the numeric weight of participants in a wrestling competition to categories such as Heavyweight, Lightweight, and Flyweight.

These techniques might be combined to derive a new feature. Ex: A categorical ranking of country by population might be calculated by rounding `log(population)` to the nearest integer.

PARTICIPATION ACTIVITY 5.5.3: Deriving data.



Animation content:

Step 1: A dataset appears with columns Country, Continent, GDP, Population, and EducationYears.
Step 2: PerCapitaGDP is added as a new column.
Step 3: EducationYears is highlighted.
Step 4: EducationLevel is added as a new column.

Animation captions:

1. GDP (Gross Domestic Product) measures total goods and services produced by a country over one year in US dollars.
2. PerCapitaGDP is a derived feature computed as GDP/Population.
3. EducationYears measures average years of education for the population of each country.
4. EducationLevel is a categorical feature derived from EducationYears.

PARTICIPATION ACTIVITY 5.5.4: Data wrangling activities.

Map the data wrangling step to the corresponding activities.

If unable to drag and drop, refresh the page.

Cleaning data **Enriching data** **Structuring data**

	Formatting, scaling, and unpacking
	Deriving and appending
	Discarding and imputing

Reset

Enriching data with Python

`pandas` has many data enriching methods. Selected methods that append and derive data are described in the table below. The table includes all required parameters and important optional parameters but excludes infrequently used optional parameters.

`df.merge()` emulates a relational database join. Relational joins merge two tables by specifying join columns in each table. The join columns correspond to the matching feature described in Appending data, above.

A relational join merges rows that have matching join column values. Relational joins can be executed in several ways, including inner, outer, left, and right joins. These join types specify how to handle rows that do not have matching join column values. Inner, outer, left, and right joins are described in detail elsewhere in this material.

Table 5.5.2: Python data enriching methods.

Method	Parameters	Description

`pd.concat()`	`objs`
	`axis=0`
	`join='outer'`
	`ignore_index=False`
		Appends dataframes specified in `objs` parameter. Appends rows if `axis=0` or columns if `axis=1`. `join` specifies whether to perform an `'outer'` or `'inner'` join. Resulting index values are unchanged if `ignore_index=False` or renumbered if `ignore_index=True`.

`df.apply()`	`func` `axis=0`	Applies the function specified in `func` parameter to a dataframe `df`. Applies function to each column if `axis=0` or to each row if `axis=1`. Returns a Series or DataFrame.

`df.insert()`	`loc`
	`column`
	`value`
		Inserts a column to `df`. `loc` specifies the integer position of the new column. `column` specifies a string or numeric column label. `value` specifies column values as a Scalar or Series.

`df.merge()`	`right`
	`how='inner'`
	`on=None`
	`sort=False`
		Joins `df` with the `right` dataframe. `how` specifies whether to perform a `'left'`, `'right'`, `'outer'`, or `'inner'` join. `on` specifies join column labels, which must appear in both dataframes. If `on=None`, all matching labels become join columns. `sort=True` sorts rows on the join columns.

Enriching data with Python.

Full screen

The Python code below reads and appends the datasets in the Appending data animation, above.

- Click the double right arrow icon to restart the kernel and run all cells.
- Note the `merge()` parameter `how='outer'`. Verify that all countries in either input dataset (Bangladesh, Brazil, China, India, Norway, and United States) appear in the result.
- Change the parameter to `how='inner'` and execute the statement. Verify that only countries in both datasets (Bangladesh, China, India, and United States) appear.
- Change the parameter to `how='left'` and execute the statement. Verify that only countries in the left dataset (Bangladesh, China, India, Norway, and United States) appear.
- Change the parameter to `how='right'` and execute the statement. Verify that only countries in the right dataset (Bangladesh, Brazil, China, India, and United States) appear.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

©zyBooks 03/31/24 10:45 2087217
Biniam abebe
UNTADTA5340OrhanSpring8wk22024

PARTICIPATION ACTIVITY 5.5.5: Appending data in Python.

Consider the two dataframes below.

`cars1`

Manufacturer	Model	Drive	EngineLiters
Audi	A4	All	2.0
BMW	328 Ci	Rear	3.6
Chevrolet	Malibu	Front	3.6
Ford	Mustang	Rear	5.0

`cars2`

Manufacturer	Model	Drive	Cylinders
Audi	A4	All	4
BMW	328 Ci	Rear	6
Chevrolet	Malibu	Front	6
Rolls-Royce	Ghost	Rear	12

1) How many instances will be in the resulting dataframe?

```
cars1.merge(cars2,
how='inner')
```

- 3
- 4
- 5

2) How many instances will be in the resulting dataframe?

```
cars1.merge(cars2,
how='outer')
```

- 3
- 4
- 5

3) Which feature will have a missing value in the resulting dataframe?

```
cars1.merge(cars2,
how='left')
```

- Cylinders
- Drive
- EngineLiters

CHALLENGE ACTIVITY 5.5.1: Enriching data.

CHALLENGE ACTIVITY | 5.5.2: Enriching data using pandas.

Start

These datasets contain information on characteristics of a variety of mammals.

- Append the data from animalsRight to animalsLeft using `dataframe.merge()` with the parameters `how='right' sort=True`.

The code provided contains all imports, loads the datasets, and displays the enriched dataframe.

main.py | mammals_append_left.csv | mammals_append_right.csv

```
# Import packages
import pandas as pd

# Load the first dataset
animalsLeft = pd.read_csv('mammals_append_left.csv')

# Load the second dataset
animalsRight = pd.read_csv('mammals_append_right.csv')

# Join the first and second datasets using the parameters how='right' and sort=True
enriched = # Your code goes here

# Print enriched dataframe
print(enriched)
```

1 | 2

Check | Next level

5.6 Case study: Diamond prices

Learning goals

- Use data manipulation to compute descriptive statistics.
- Use data restructuring to normalize and standardize features.
- Use data cleaning to identify missing and unusual values.
- Use data enrichment to calculate new features.

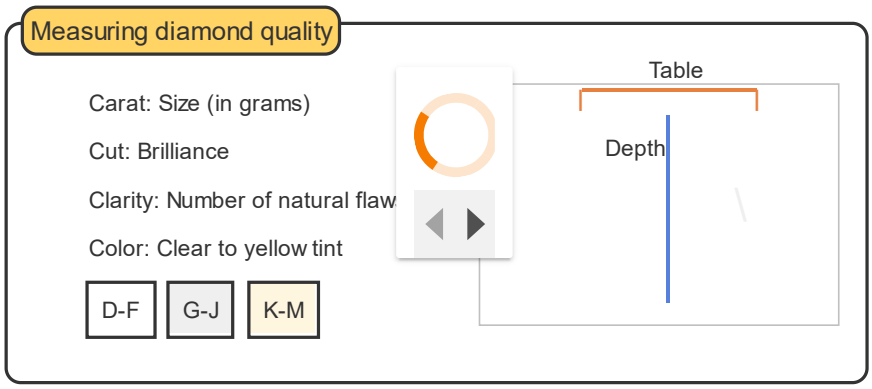
- Use data wrangling methods from Python to explore a dataset.

Diamond quality and price

Data scientists working for retailers or manufacturers use historical sales data to track demand and set prices for products. For luxury goods like diamonds, several factors affect the price of a particular item. Diamonds that are considered higher quality, like larger or more brilliant diamonds, sell for a higher price than lower quality diamonds.

What factors have the greatest influence on diamond prices? The diamonds dataset contains measurements on a sample of nearly 54,000 diamonds, including the retail price. Since the dataset is so large, data wrangling is likely needed.

PARTICIPATION ACTIVITY 5.6.1: Measuring diamond quality.



Animation content:

An image of a diamond appears. Components of the diamond are labeled as described in the captions.

Animation captions:

1. The diamonds dataset contains features describing the quality of a diamond. Each feature may have an impact on the price of a diamond.
2. Carat describes the size, or weight, of a diamond. One carat equals 0.2 grams.
3. Cut describes the brilliance of a diamond, or how much light shines through it. Cut may be Ideal, Premium, Very Good, Good, Fair, or Poor.
4. Clarity measures natural flaws like chips or scratches, called inclusions. Diamonds with no inclusions are flawless (FL), and diamonds with obvious chips are included (I).
5. Color ranges from D-M. Colorless diamonds are rated D-F, near colorless diamonds are rated G-J, and diamonds with a faint yellow tint are rated K-M.
6. Table refers to the diamond's top surface and is measured as a percentage of overall size.

7. Depth refers to the diamond's overall height. Like table, depth is measured as a percentage of the overall size.

PARTICIPATION ACTIVITY 5.6.2: What makes a diamond high quality?

The dataset below contains a random sample of five diamonds.

ID	Carat	Cut	Color	Clarity	Price
9359	1.00	Good	F	SI1	4586
9490	1.10	Very Good	E	SI2	4607
27	0.30	Very Good	J	VS2	357
7743	1.24	Premium	E	SI2	4278
48686	0.77	Fair	J	VS1	2005

1) Which diamond is largest?
   - 27
   - 7743
   - 48686

2) Which diamond has the least brilliance?
   - 27
   - 7743
   - 48686

3) Which diamond(s) have the clearest color?
   - 9359
   - 9490 and 7743
   - 27 and 48686

4) Which diamond is most expensive?
   - 9359
   - 9490
   - 27

Step 1: Discovering data

A data scientist should familiarize themselves with a dataset before modeling. Since the objective is exploring factors related to diamond price, descriptive statistics and pivot tables will be useful. Ex: Is the average price of colorless diamonds higher than the average price of tinted diamonds?

Discovering the diamonds data with Python. Full screen

The Python code below imports the diamonds dataset and calculates descriptive statistics and pivot tables.

- Click the double right arrow icon to restart the kernel and run all cells.

- Examine the code below.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

Data source: Hadley Wickham, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2022. "Prices of over 50,000 round cut diamonds." https://ggplot2.tidyverse.org/reference/diamonds.html

PARTICIPATION ACTIVITY 5.6.3: Discovering the diamonds data.

1) Which diamond color rating has the highest average price?

- D
- G
- J

2) Which diamond cut has the lowest average price?

- Fair
- Ideal
- Premium

3) Which group had the least diamonds?

- Fair cut/J color
- Fair cut/D color
- Ideal cut/D color

Step 2: Structuring data

During data structuring, features in a dataset are converted into a particular format or scaled using normalization or standardization. But not all features in a dataset require restructuring. In some cases, the original features have meaning. Ex: Carat is a unit for measuring the size of a diamond and should not be restructured. The actual carat value has a specific meaning.

Structuring the diamonds data with Python. Full screen

The Python code below imports the diamonds dataset, standardizes or normalizes selected features, and plots the restructured features.

- Click the double right arrow icon to restart the kernel and run all cells.
- Examine the code below.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

PARTICIPATION ACTIVITY 5.6.4: Restructuring the diamonds data.

1) Which restructuring technique, if any, should be applied to diamond price?
   - ○ Normalization
   - ○ Standardization
   - ○ No restructuring

2) The maximum dimensions - width, length, and height - of each diamond is measured in millimeters. Which restructuring technique, if any, should be applied to diamond width?
   - ○ Normalization
   - ○ Standardization
   - ○ No restructuring

3) The table describes the relationship between the size of a diamond's top edge and the rest of the diamond. The ideal diamond table is considered to be 60. Which restructuring technique, if any, should be applied to diamond table?
   - ○ Normalization
   - ○ Standardization
   - ○ No restructuring

Step 3: Cleaning data

Missing values or outliers are removed during the data cleaning stage. But, not all missing values or outliers should be removed. Values may be missing from a dataset for a specific reason. Ex: A diamond that hasn't been sold yet may have a missing value for price. Outliers describe extremes of a distribution, and should not be removed without reason. Ex: An outlier resulting from a typo should be removed. But, an outlier for carat that truly represents a much larger diamond should be kept in the dataset.

Cleaning the diamonds data with Python. Full screen

The Python code below imports the diamonds dataset and searches for missing values and outliers.

- Click the double right arrow icon to restart the kernel and run all cells.
- Examine the code below.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

PARTICIPATION ACTIVITY 5.6.5: Cleaning the diamonds data.

1) Which feature in the diamonds dataset contains missing values?
   - Carat
   - Depth
   - Length

2) The average depth is calculated and used to replace missing values. Which form of imputation is described?
   - Mean imputation
   - Median imputation
   - Regression imputation

3) Which feature does not have at least one outlier?
   - Carat
   - Length
   - Price

Step 4: Enriching data

New features are derived from existing ones during the data enrichment process. New features may be calculated as mathematical functions of other features. Ex: The ratio of length divided by the width may provide some information about a diamond's shape.

Data enrichment is not limited to numerical features. Categorical features may be enriched by combining or splitting levels. Ex: Diamond clarity is rated on a scale with 11 levels, which are often combined into a smaller set of six levels.

Enriching the diamonds data with Python.

Full screen

The Python code below imports the diamonds dataset and creates new features.

- Click the double right arrow icon to restart the kernel and run all cells.
- Examine the code below.

Your server is starting up.

You will be redirected automatically when it's ready for you.

Event log

PARTICIPATION ACTIVITY 5.6.6: Enriching the diamonds data.

The diamonds dataset does not contain a feature describing the shape of the diamond. But the ratio of length divided by width gives some insights.

Diamond shape	Description
Round or brilliant	Circular shaped
Princess	Square with pointed edges
Cushion	Square with rounded edges
Oval	Elongated circle with round edges
Marquise	Oval with points at the top and bottom
Pear	Elongated, with a point at the top and rounded at the bottom

1) What ratio would a cushion-shaped diamond have?
   - Less than 1
   - Near 1
   - Greater than 1

2) What ratio would a pear-shaped diamond have?
   - Less than 1
   - Near 1
   - Greater than 1

3) What shape does a diamond with ratio 2 most likely have?
   - Cushion
   - Oval
   - Round

5.7 LAB: Cleaning data using dropna() and fillna()

The `hmeq_small` dataset contains information on 5960 home equity loans, including 7 features on the characteristics of the loan.

- Load the data set `hmeq_small.csv` as a data frame.
- Create a new data frame with all the rows with missing data deleted.
- Create a second data frame with all missing data filled in with the mean value of the column.
- Find the means of the columns for both new data frames.

Ex: Using only the first hundred rows, found in `hmeq_sample.csv`, the output is:

```
Means for hmeqDelete are  LOAN        3208.333333
MORTDUE    67495.958333
VALUE      82529.125000
YOJ            8.500000
CLAGE        144.749455
```

```
CLNO          16.583333
DEBTINC       33.052122
dtype: float64
Means for hmeqReplace are  LOAN        3045.918367
MORTDUE    49386.494253
VALUE      64033.483871
YOJ            8.179775
CLAGE        140.209320
CLNO          15.586957
DEBTINC       30.947152
dtype: float64
```


537150.4174434.qx3zqy7

LAB ACTIVITY 5.7.1: LAB: Cleaning data using dropna() and fillna() 0 / 1

main.py

```
1 Loading latest submission...
```

Develop mode | **Submit mode**

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Enter program input (optional)

If your code requires input values, provide them here.

Run program

Input (from above) → **main.py** (Your program) → Output (shown below)

Program output displayed here

Coding trail of your work What is this?

Retrieving signature