## DATA PREPARATION (CLEANSING, INTEGRATING, TRANSFORMING DATA)

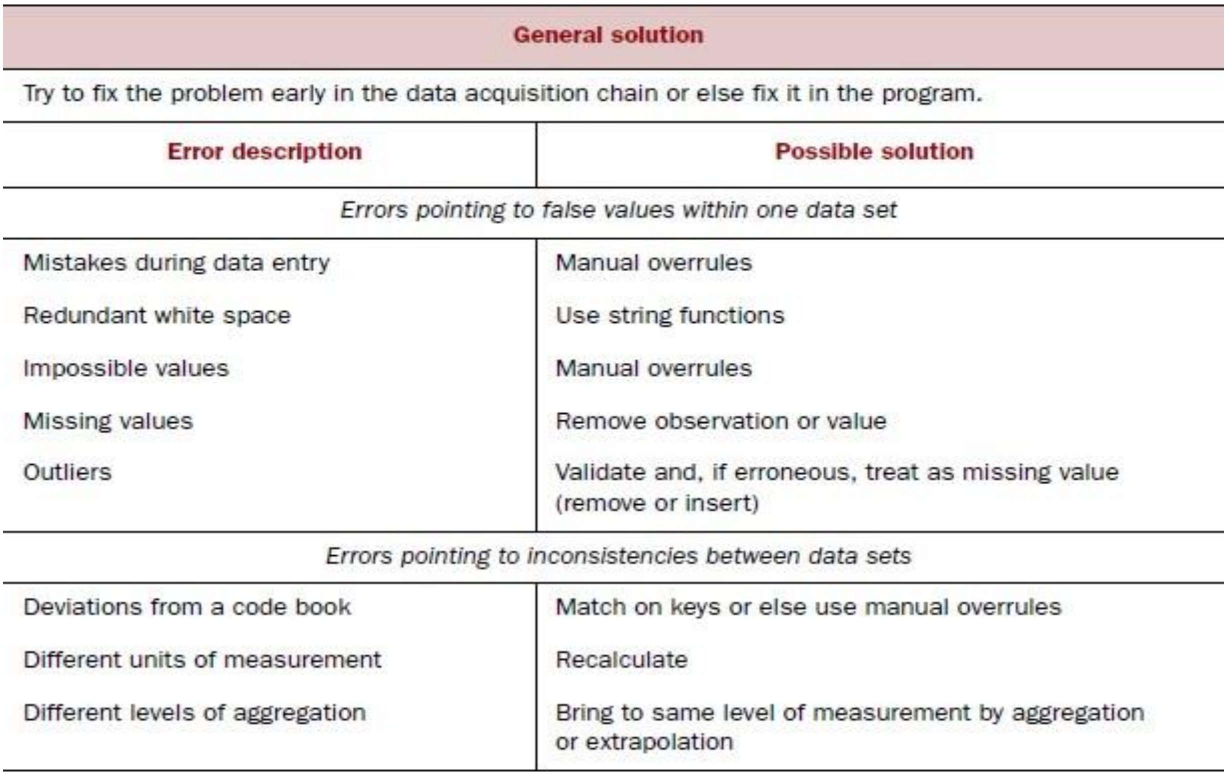
Your model needs the data in a specific format, so data transformation will always come into play. It's a good habit to correct data errors as early on in the process as possible. However, this isn't always possible in a realistic setting, so you'll need to take corrective actions in your program.

### Cleansing data

Data cleansing is a sub process of the data science process that focuses on removing errors in your data so your data becomes a true and consistent representation of the processes it originates from.

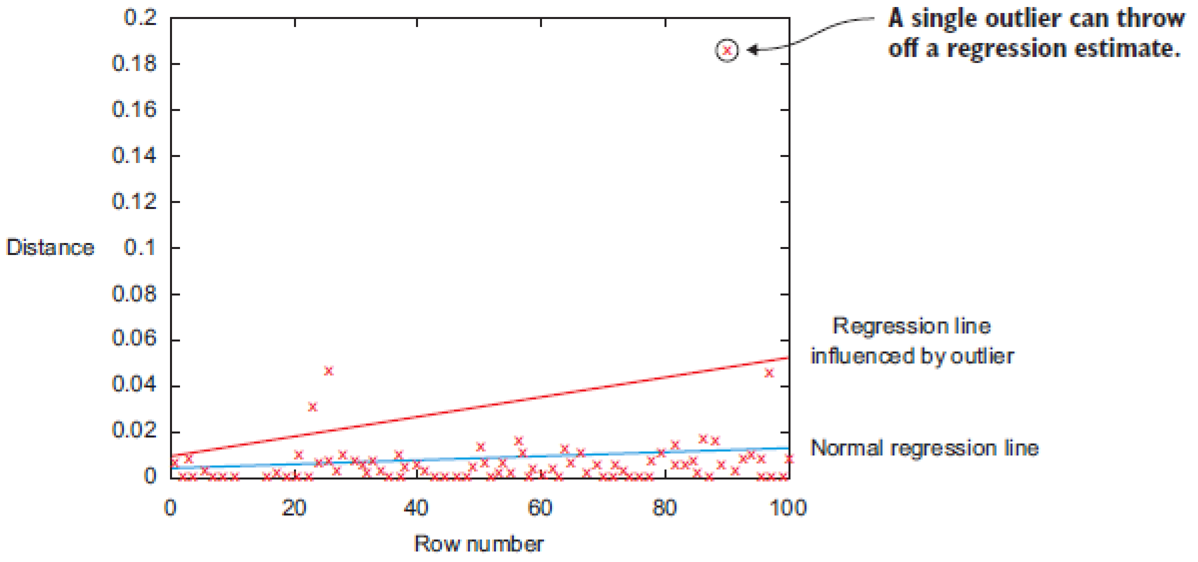
- The first type is the interpretation error, such as when you take the value in your data for granted, like saying that a person's age is greater than 300 years.
- The second type of error points to inconsistencies between data sources or against your company's standardized values. An example of this class of errors is putting "Female" in one table and "F" in another when they represent the same thing: that the person is female.

### Overview of common errors



| General solution | |
|---|---|
| Try to fix the problem early in the data acquisition chain or else fix it in the program. | |
| **Error description** | **Possible solution** |
| *Errors pointing to false values within one data set* | |
| Mistakes during data entry | Manual overrules |
| Redundant white space | Use string functions |
| Impossible values | Manual overrules |
| Missing values | Remove observation or value |
| Outliers | Validate and, if erroneous, treat as missing value (remove or insert) |
| *Errors pointing to inconsistencies between data sets* | |
| Deviations from a code book | Match on keys or else use manual overrules |
| Different units of measurement | Recalculate |
| Different levels of aggregation | Bring to same level of measurement by aggregation or extrapolation |

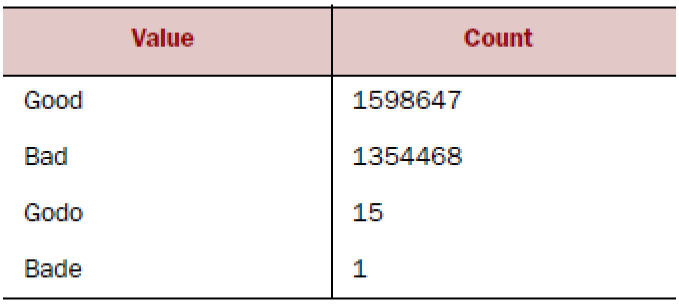
Sometimes you'll use more advanced methods, such as simple modeling, to find and identify data errors; diagnostic plots can be especially insightful. For example, in figure we use a measure to identify data points that seem out of place. We do a regression to get acquainted with the data and detect the influence of individual observations on the regression line.



A single outlier can throw off a regression estimate.

## Data Entry Errors

- Data collection and data entry are error-prone processes. They often require human intervention, and introduce an error into the chain.
- Data collected by machines or computers isn't free from errors. Errors can arise from human sloppiness, whereas others are due to machine or hardware failure.
- Detecting data errors when the variables you study don't have many classes can be done by tabulating the data with counts.
- When you have a variable that can take only two values: "Good" and "Bad", you can create a frequency table and see if those are truly the only two values present. In table the values "Godo" and "Bade" point out something went wrong in at least 16 cases.



| Value | Count |
|---|---|
| Good | 1598647 |
| Bad | 1354468 |
| Godo | 15 |
| Bade | 1 |

Most errors of this type are easy to fix with simple assignment statements and if-thenelse rules:
if x == "Godo":
x = "Good" if x == "Bade":
x = "Bad"

## Redundant Whitespace

- Whitespaces tend to be hard to detect but cause errors like other redundant characters would.
- The whitespace cause the miss match in the string such as "FR " – "FR", dropping the observations that couldn't be matched.

- If you know to watch out for them, fixing redundant whitespaces is luckily easy enough in most programming languages. They all provide string functions that will remove the leading and trailing whitespaces. For instance, in Python you can use the strip() function to remove leading and trailing spaces.

### Fixing Capital Letter Mismatches

Capital letter mismatches are common. Most programming languages make a distinction between "Brazil" and "brazil".

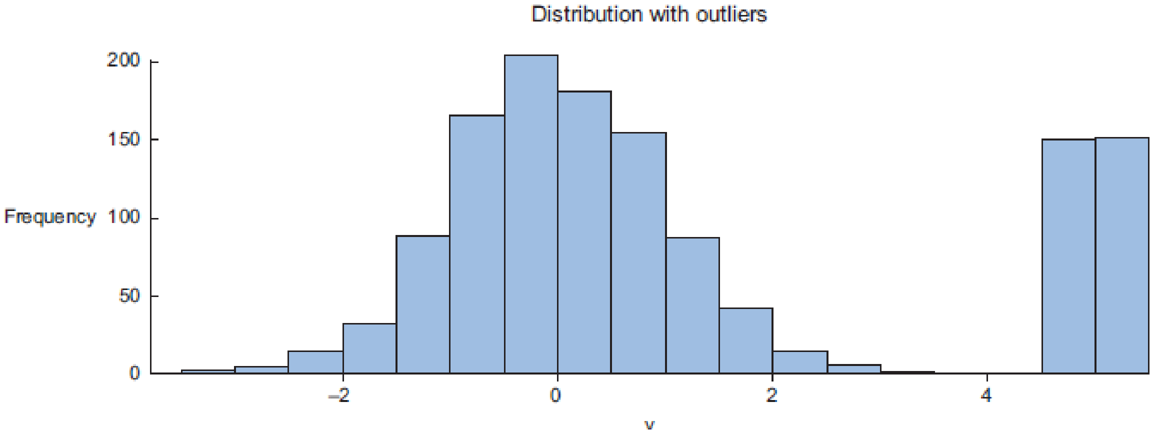
In this case you can solve the problem by applying a function that returns both strings in lowercase, such as .lower() in Python. "Brazil".lower() == "brazil".lower() should result in true.

### Impossible Values and Sanity Checks

Here you check the value against physically or theoretically impossible values such as people taller than 3 meters or someone with an age of 299 years. Sanity checks can be directly expressed with rules: check = 0 <= age <= 120

### Outliers

An outlier is an observation that seems to be distant from other observations or, more specifically, one observation that follows a different logic or generative process than the other observations. The easiest way to find outliers is to use a plot or a table with the minimum and maximum values. The plot on the top shows no outliers, whereas the plot on the bottom shows possible outliers on the upper side when a normal distribution is expected.



### Dealing with Missing Values

Missing values aren't necessarily wrong, but you still need to handle them separately; certain modeling techniques can't handle missing values. They might be an indicator that something went wrong in your data collection or that an error happened in the ETL process. Common techniques data scientists use is listed in table.

### Integrating data

Your data comes from several different places, and in this substep we focus on integrating these different sources. Data varies in size, type, and structure, ranging from databases and Excel files to text documents.

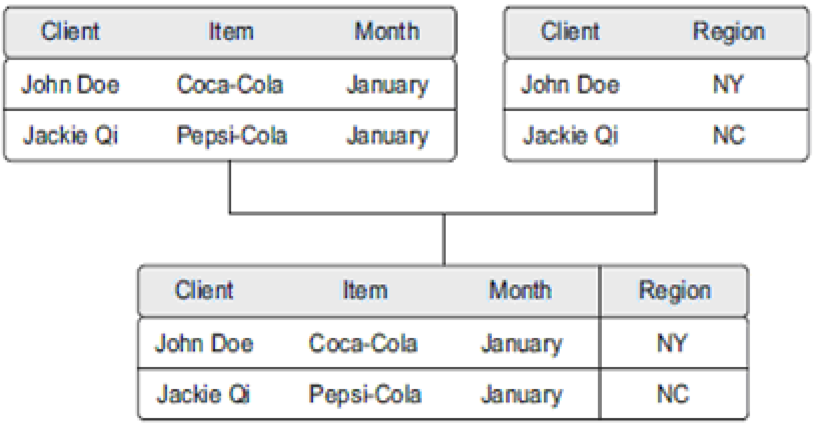
### The Different Ways of Combining Data

You can perform two operations to combine information from different data sets.

- Joining
- Appending or stacking

## Joining Tables

- Joining tables allows you to combine the information of one observation found in one table with the information that you find in another table. The focus is on enriching a single observation.
- Let's say that the first table contains information about the purchases of a customer and the other table contains information about the region where your customer lives.
- Joining the tables allows you to combine the information so that you can use it for your model, as shown in figure.



| Client | Item | Month |
|---|---|---|
| John Doe | Coca-Cola | January |
| Jackie Qi | Pepsi-Cola | January |

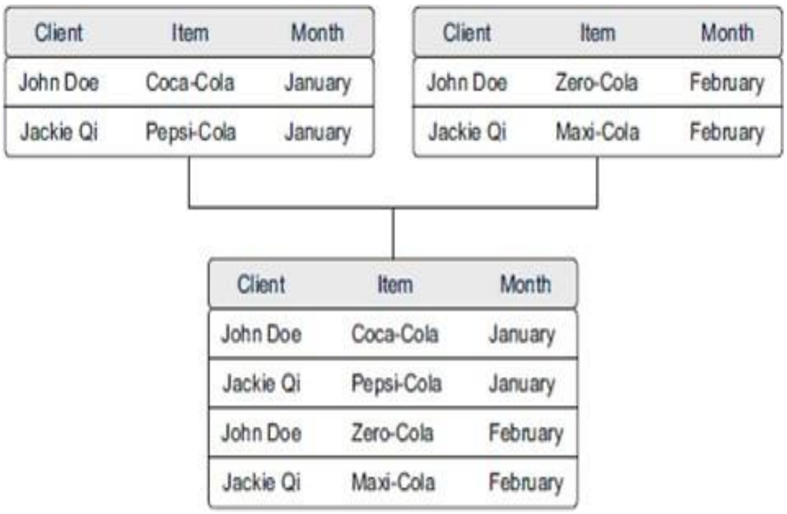
| Client | Region |
|---|---|
| John Doe | NY |
| Jackie Qi | NC |

| Client | Item | Month | Region |
|---|---|---|---|
| John Doe | Coca-Cola | January | NY |
| Jackie Qi | Pepsi-Cola | January | NC |

To join tables, you use variables that represent the same object in both tables, such as a date, a country name, or a Social Security number. These common fields are known as keys. When these keys also uniquely define the records in the table, they are called *primary keys*. The number of resulting rows in the output table depends on the exact join type that you use

## Appending Tables

- Appending or stacking tables is effectively adding observations from one table to another table.
- One table contains the observations from the month January and the second table contains observations from the month February. The result of appending these tables is a larger one with the observations from January as well as February.



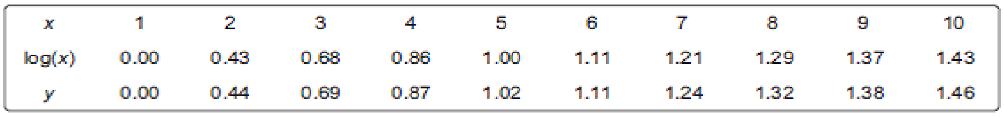
| Client | Item | Month |
|---|---|---|
| John Doe | Coca-Cola | January |
| Jackie Qi | Pepsi-Cola | January |

| Client | Item | Month |
|---|---|---|
| John Doe | Zero-Cola | February |
| Jackie Qi | Maxi-Cola | February |

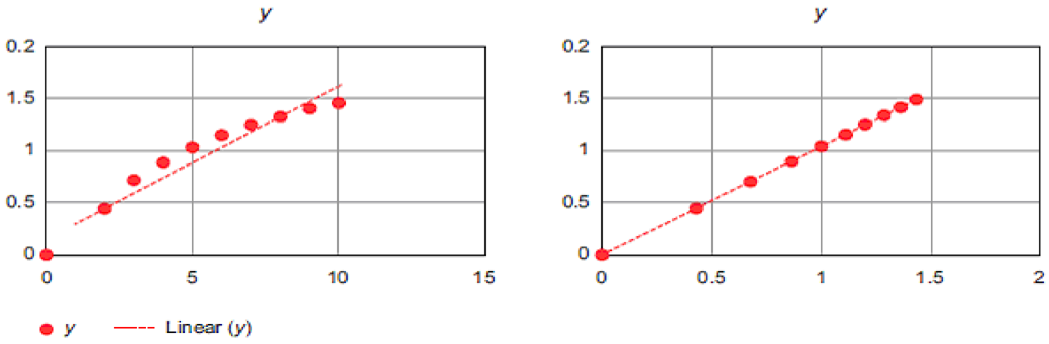
| Client | Item | Month |
|---|---|---|
| John Doe | Coca-Cola | January |
| Jackie Qi | Pepsi-Cola | January |
| John Doe | Zero-Cola | February |
| Jackie Qi | Maxi-Cola | February |

## Transforming data

Certain models require their data to be in a certain shape. Transforming your data so it takes a suitable form for data modeling. Relationships between an input variable and an output variable aren't always linear. Take, for instance, a relationship of the form *y* = *aebx*. Taking the log of the independent variables simplifies the estimation problem dramatically. Transforming the input variables greatly simplifies the estimation problem. Other times you might want to combine two variables into a new variable.



| *x* | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| log(*x*) | 0.00 | 0.43 | 0.68 | 0.86 | 1.00 | 1.11 | 1.21 | 1.29 | 1.37 | 1.43 |
| *y* | 0.00 | 0.44 | 0.69 | 0.87 | 1.02 | 1.11 | 1.24 | 1.32 | 1.38 | 1.46 |



**Transforming x to log x makes the relationship between x and y linear (right), compared with the non-log x (left).**

## Reducing the Number of Variables

- ➢ Having too many variables in your model makes the model difficult to handle, and certain techniques don't perform well when you overload them with too many input variables. For instance, all the techniques based on a Euclidean distance perform well only up to 10 variables.
- ➢ Data scientists use special methods to reduce the number of variables but retain the maximum amount of data.

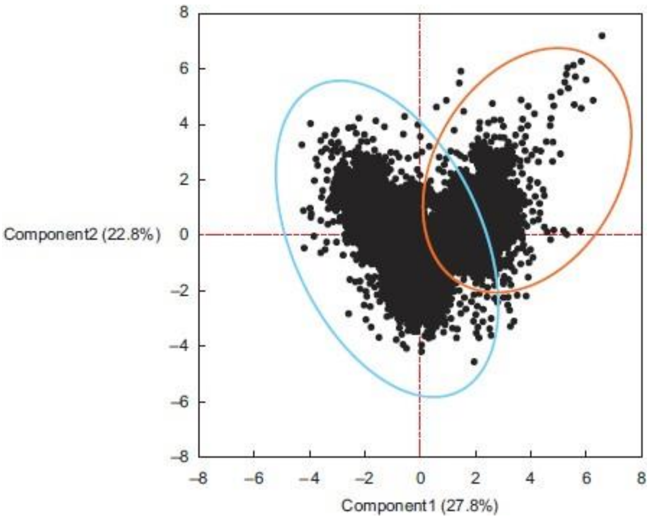
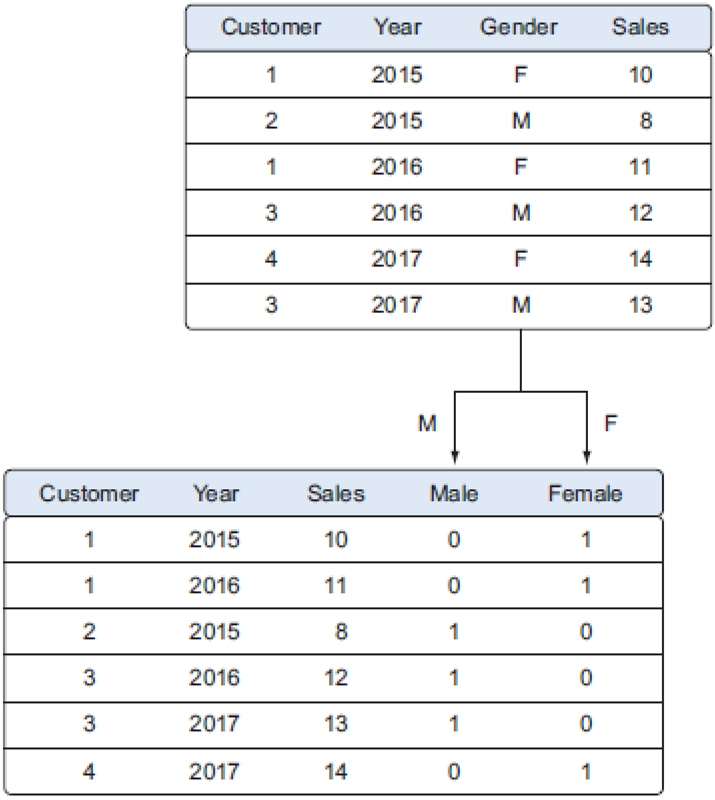


Figure shows how reducing the number of variables makes it easier to understand the key values. It also shows how two variables account for 50.6% of the variation within the data set (component1 = 27.8% + component2 = 22.8%). These variables, called "component1" and "component2," are both combinations of the original variables. They're the *principal components* of the underlying data structure.

### Turning Variables into Dummies

- Dummy variables can only take two values: true(1) or false(0). They're used to indicate the absence of a categorical effect that may explain the observation.
- In this case you'll make separate columns for the classes stored in one variable and indicate it with 1 if the class is present and 0 otherwise.
- An example is turning one column named Weekdays into the columns Monday through Sunday. You use an indicator to show if the observation was on a Monday; you put 1 on Monday and 0 elsewhere.
- Turning variables into dummies is a technique that's used in modeling and is popular with, but not exclusive to, economists.



| Customer | Year | Gender | Sales |
|---|---|---|---|
| 1 | 2015 | F | 10 |
| 2 | 2015 | M | 8 |
| 1 | 2016 | F | 11 |
| 3 | 2016 | M | 12 |
| 4 | 2017 | F | 14 |
| 3 | 2017 | M | 13 |

| Customer | Year | Sales | Male | Female |
|---|---|---|---|---|
| 1 | 2015 | 10 | 0 | 1 |
| 1 | 2016 | 11 | 0 | 1 |
| 2 | 2015 | 8 | 1 | 0 |
| 3 | 2016 | 12 | 1 | 0 |
| 3 | 2017 | 13 | 1 | 0 |
| 4 | 2017 | 14 | 0 | 1 |

Figure. Turning variables into dummies is a data transformation that breaks a variable that has multiple classes into multiple variables, each having only two possible values: 0 or 1