

Visualization: Scatter Plots & Line Plots

Prof. Abby Stylianou

Logistics

- Quiz on “Using Metadata to Find Paul Revere” Thursday
 - Code snippets are in R but should be readable
- Assignment 2 out on Thursday, due the following Thursday
 - (could be Tuesday – Tuesday, depending on how much we get through)

Tables

- A Table is a sequence of labeled columns
- Each row represents one individual
- Data within a column represents one attribute of the individuals

The diagram illustrates a table with three columns: Name, Code, and Area (m2). The first row contains 'California', 'CA', and '163696'. The second row contains 'Nevada', 'NV', and '110567'. Annotations include a 'Label' box pointing to the 'Code' header, a 'Row' box pointing to the 'Nevada' row, and a 'Column' box pointing to the 'Code' column. A blue box highlights the 'CA' and 'NV' cells, and another blue box highlights the 'Nevada' and 'NV' cells.

Name	Code	Area (m2)
California	CA	163696
Nevada	NV	110567

Table Methods

- Creating and extending tables:
 - `Table().with-column` and `Table.read_table(csv_file)`

Table Methods

- Creating and extending tables:
 - `Table().with-column` and `Table.read_table(csv_file)`
- Finding the size:
 - `num_rows` and `num_columns`

Table Methods

- Creating and extending tables:
 - `Table().with-column` and `Table.read_table(csv_file)`
- Finding the size:
 - `num_rows` and `num_columns`
- Referring to columns: labels, relabeling, and indices
 - `labels` and `relabelled`; column indices start at 0

Table Methods

- Creating and extending tables:
 - `Table().with-column` and `Table.read_table(csv_file)`
- Finding the size:
 - `num_rows` and `num_columns`
- Referring to columns: labels, relabeling, and indices
 - `labels` and `relabelled`; column indices start at 0
- Accessing data in a column
 - `column` takes a label or index and returns an array

Table Methods

- Creating and extending tables:
 - `Table().with-column` and `Table.read_table(csv_file)`
- Finding the size:
 - `num_rows` and `num_columns`
- Referring to columns: labels, relabeling, and indices
 - `labels` and `relabelled`; column indices start at 0
- Accessing data in a column
 - `column` takes a label or index and returns an array
- Using array methods to work with data in columns
 - `item`, `sum`, `min`, `max`, etc.

Table Methods

- Creating and extending tables:
 - `Table().with-column` and `Table.read_table(csv_file)`
- Finding the size:
 - `num_rows` and `num_columns`
- Referring to columns: labels, relabeling, and indices
 - `labels` and `relabelled`; column indices start at 0
- Accessing data in a column
 - `column` takes a label or index and returns an array
- Using array methods to work with data in columns
 - `item`, `sum`, `min`, `max`, etc.
- Creating new tables containing some of the original columns:
 - `select`, `drop`

Manipulating Rows

- `t.sort (column)` sorts the rows in increasing order

Manipulating Rows

- `t.sort(column)` sorts the rows in increasing order
- `t.take(row_numbers)` keeps the numbered rows
 - Each row has an index, starting at 0

Manipulating Rows

- `t.sort(column)` sorts the rows in increasing order
- `t.take(row_numbers)` keeps the numbered rows
 - Each row has an index, starting at 0
- `t.where(column, are.condition)` keeps all rows for which a column's value satisfies a condition
 - `are.equal_to(5)`, `are.above(20)`, `are.below(10)`,
`are.between(30, 38)`, etc
 - <http://data8.org/datascience/predicates.html>

Manipulating Rows

- `t.sort(column)` sorts the rows in increasing order
- `t.take(row_numbers)` keeps the numbered rows
 - Each row has an index, starting at 0
- `t.where(column, are.condition)` keeps all rows for which a column's value satisfies a condition
 - `are.equal_to(5)`, `are.above(20)`, `are.below(10)`, `are.between(30, 38)`, etc
 - <http://data8.org/datascience/predicates.html>
- `t.where(column, value)` keeps all rows for which a column's value equals some particular value

Manipulating Rows

- `t.sort(column)` sorts the rows in increasing order
- `t.take(row_numbers)` keeps the numbered rows
 - Each row has an index, starting at 0
- `t.where(column, are.condition)` keeps all rows for which a column's value satisfies a condition
 - `are.equal_to(5)`, `are.above(20)`, `are.below(10)`, `are.between(30, 38)`, etc
 - <http://data8.org/datascience/predicates.html>
- `t.where(column, value)` keeps all rows for which a column's value equals some particular value
- `t.with_row(list)` makes a new table that has another row with values in *list*

Tables

- We've worked with small datasets (and some bigger ones last class!)
- No matter how well a large table is organized, it can be difficult to interpret

The diagram illustrates the components of a table using a dataset of US states. A blue speech bubble labeled 'Label' points to the 'Code' header. A blue rounded rectangle highlights the 'CA' and 'NV' entries in the 'Code' column, with a blue speech bubble labeled 'Column' pointing to it. Another blue rounded rectangle highlights the entire row for 'Nevada', with a blue speech bubble labeled 'Row' pointing to it.

Name	Code	Area (m2)
California	CA	163696
Nevada	NV	110567

The Decennial Census

- Every 10 years, the Census Bureau counts # of people in the US
- In between censuses, the Bureau estimates how many people there are each year
- Article 1, Section 2 of the United States Constitution:
 - “Representatives and direct Taxes shall be apportioned among the several States ... according to their respective numbers ...”

The Decennial Census

- Demo!

cs1070.com → Demos → new notebook

Census Table Description

- Values have column-dependent interpretations
 - The SEX column: 1 = male, 2 = Female
- In this table, some rows are sums of other rows
 - The SEX column: 0 is Total (male + female)
 - The AGE column: 999 is Total (all ages)
- Numeric codes are often used for storage efficiency
- Values in a column may have the same type but not be directly comparable (AGE 999 vs AGE 12, AGE 100 – everyone > 100)

Growth Rate

- Growth rate = g (for example 3%, or 0.03)
- Initial value x , final value y after t periods of time

Value after 1 period	=	$x + xg$	=	$x * (1+g)$
2 periods	=	$x(1+g)(1+g)$	=	$x * (1+g) ** 2$
t periods	=	y	=	$x * (1+g) ** t$

Computing the Growth Rate

$$(1+g)^t = y/x$$

$$1 + g = (y/x)^{1/t}$$

$$g = (y/x)^{1/t} - 1$$

Types of Data

- All values in a column should be both the same type *and* comparable to each other in some way

Types of Data

- All values in a column should be both the same type *and* comparable to each other in some way
 - **Numerical** – each value is from a numerical scale
 - Numerical measurements are ordered
 - Differences are meaningful

Types of Data

- All values in a column should be both the same type *and* comparable to each other in some way
 - **Numerical** – each value is from a numerical scale
 - Numerical measurements are ordered
 - Differences are meaningful
 - **Categorical** – each value is from a fixed inventory
 - May or may not have an ordering
 - Categories are the same or different

“Numerical” Data

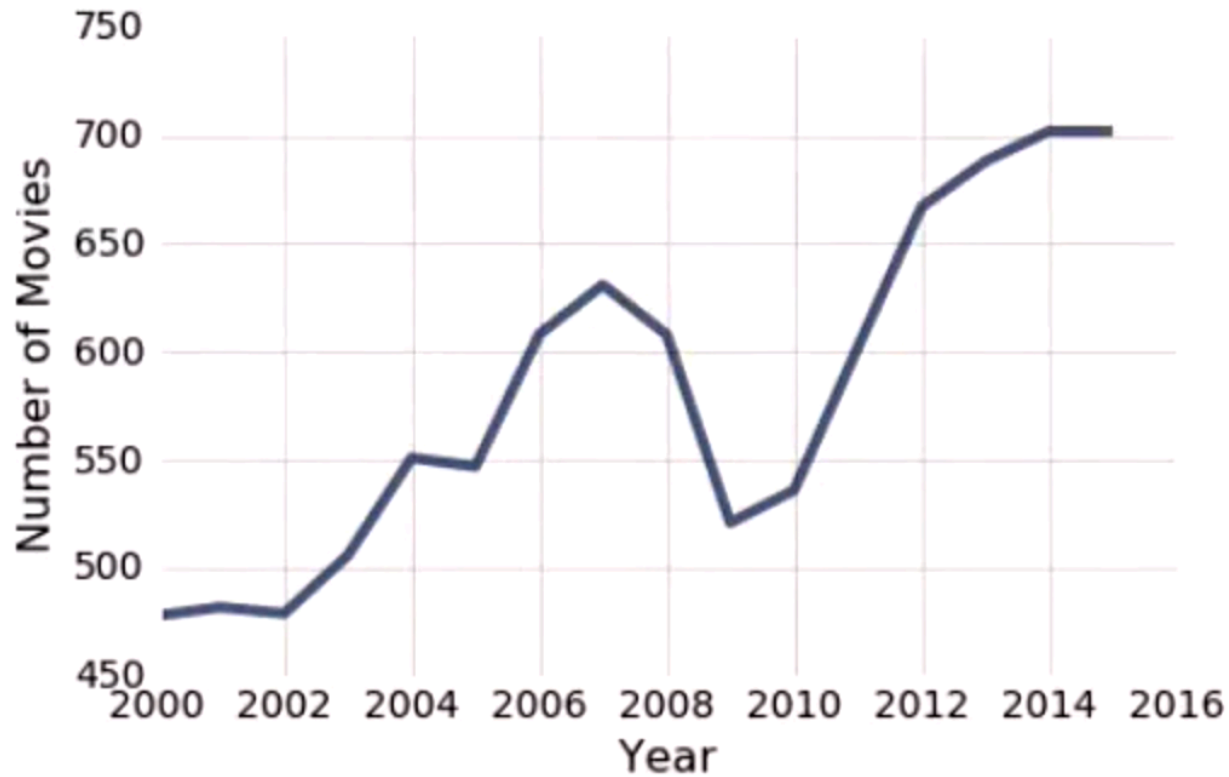
- Just because the values are numbers doesn't mean the variable is numerical
 - Census example had numerical SEX code (0, 1 and 2)
 - It doesn't make sense to perform arithmetic on these “numbers”, e.g., $1 - 0$ or $(0+1+2)/3$ are meaningless

“Numerical” Data

- Just because the values are numbers doesn't mean the variable is numerical
 - Census example had numerical SEX code (0, 1 and 2)
 - It doesn't make sense to perform arithmetic on these “numbers”, e.g., $1 - 0$ or $(0+1+2)/3$ are meaningless
- The variable SEX is still categorical even though they're numbers

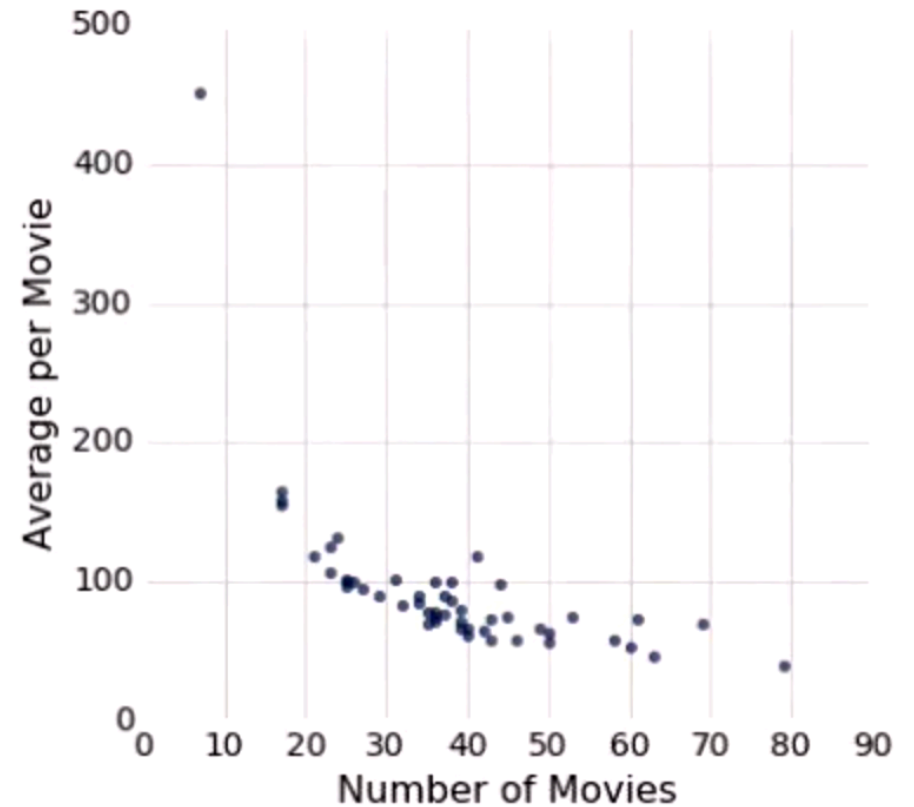
Plotting Two Numerical Variables

Line graph: `plot`



How something changes as the X-axis changes
(often chronologically)

Scatter plot: `scatter`



Comparing two numerical
variables

Anthony Daniels
(actor)

