DSC 10, Spring 2018
Lecture 4

Tables

sites.google.com/eng.ucsd.edu/dsc-10-spring-2018

Credit: Anindita Adhikari and John DeNero

Announcements

- Lab 2 due Wednesday 11:59pm

- Guest Speaker tomorrow 6pm, Mandeville B210

Dr. Aaron Fraenkel, Sr. Machine Learning Scientist, Amazon

Bias, Fairness, and Interpretability in Machine Learning

“Machine Learning occasionally makes news for producing results perceived as biased, unfair, or just fantastically wrong. We’ll examine why these results aren’t mysterious or unexpected, as long as you ask the right questions and know the data driving your results.”

Ranges

Ranges

A range is an array of consecutive numbers

Ranges

A range is an array of consecutive numbers

- **`np.arange(end)`**:
  An array of increasing integers from 0 up to **`end`**

Ranges

A range is an array of consecutive numbers

- **`np.arange(end)`**:
  An array of increasing integers from 0 up to **`end`**
- **`np.arange(start, end)`**:
  An array of increasing integers from **`start`** up to **`end`**

Ranges

A range is an array of consecutive numbers

- **`np.arange(end)`**:
  An array of increasing integers from 0 up to **`end`**
- **`np.arange(start, end)`**:
  An array of increasing integers from **`start`** up to **`end`**
- **`np.arange(start, end, step)`**:
  A range with **`step`** between consecutive values

Ranges

A range is an array of consecutive numbers

- **`np.arange(end)`**:
  An array of increasing integers from 0 up to **`end`**
- **`np.arange(start, end)`**:
  An array of increasing integers from **`start`** up to **`end`**
- **`np.arange(start, end, step)`**:
  A range with **`step`** between consecutive values

A range always includes **`start`** but excludes **`end`**

[,)

Discussion Question

Assume you have run the following statements.

```
x = make_array(2, 3, 4)
y = np.arange(2, 3, 4)
z = np.arange(3)
```

Which line(s) will cause an error?

A. `x + y`

B. `x + z`

C. `x.item(0)+y.item(0)`

D. `x.item(1)+y.item(1)`

(Demo)

Leibniz Formula for Pi

$$\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \ldots\right)$$

```
a = np.arange(1, 12, 4)
```

```
a
```

```
a + 2
```

```
4 * sum(1/a - 1/(a+2))
```

Leibniz formula for π

Leibniz Formula for Pi

$$\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \ldots\right)$$

```
a = np.arange(1, 12, 4)
```

```
a
```

```
a + 2
```

```
4 * sum(1/a - 1/(a+2))
```

Which of the following is false?

A. `sum` is a function being applied to an array
B. `a` is an array
C. `a` is a range
D. The last line is equivalent to

```
4 * (1/a + -1/(a+2))
```

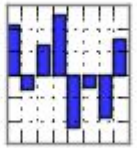
(Demo)

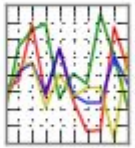
Leibniz formula for π

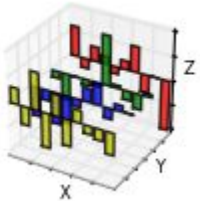
Tables

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



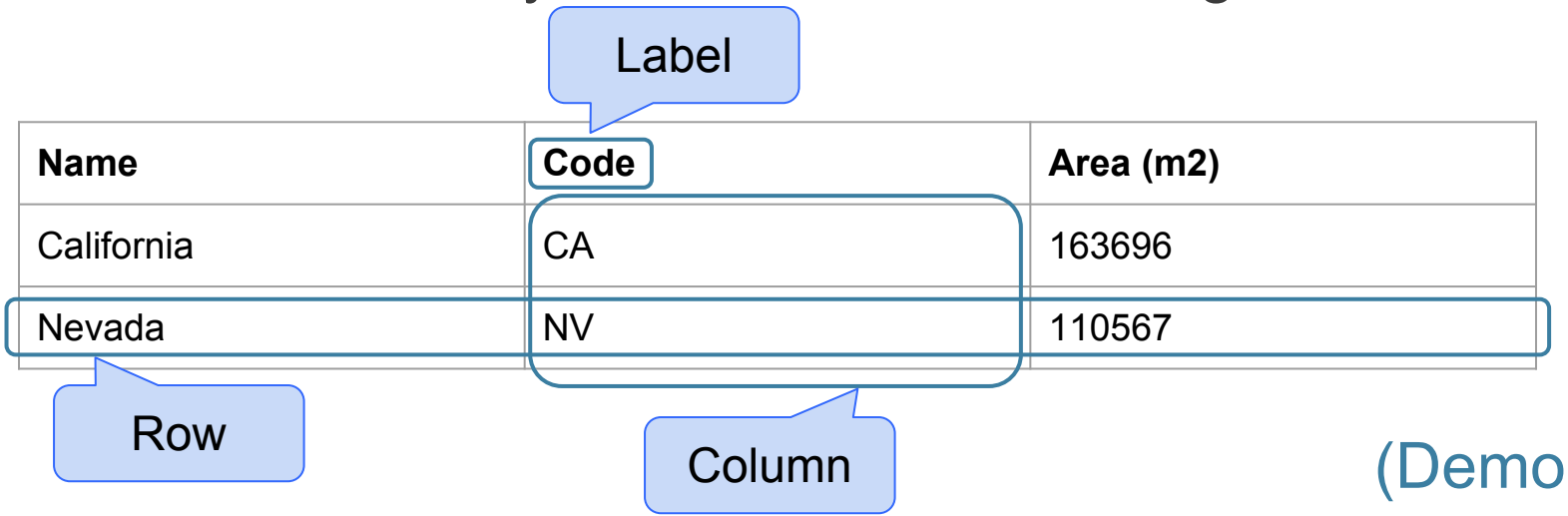




[Python Data Analysis Library]

Table Structure

- A Table is a sequence of labeled columns
- Labels are strings
- Columns are arrays, all with the same length



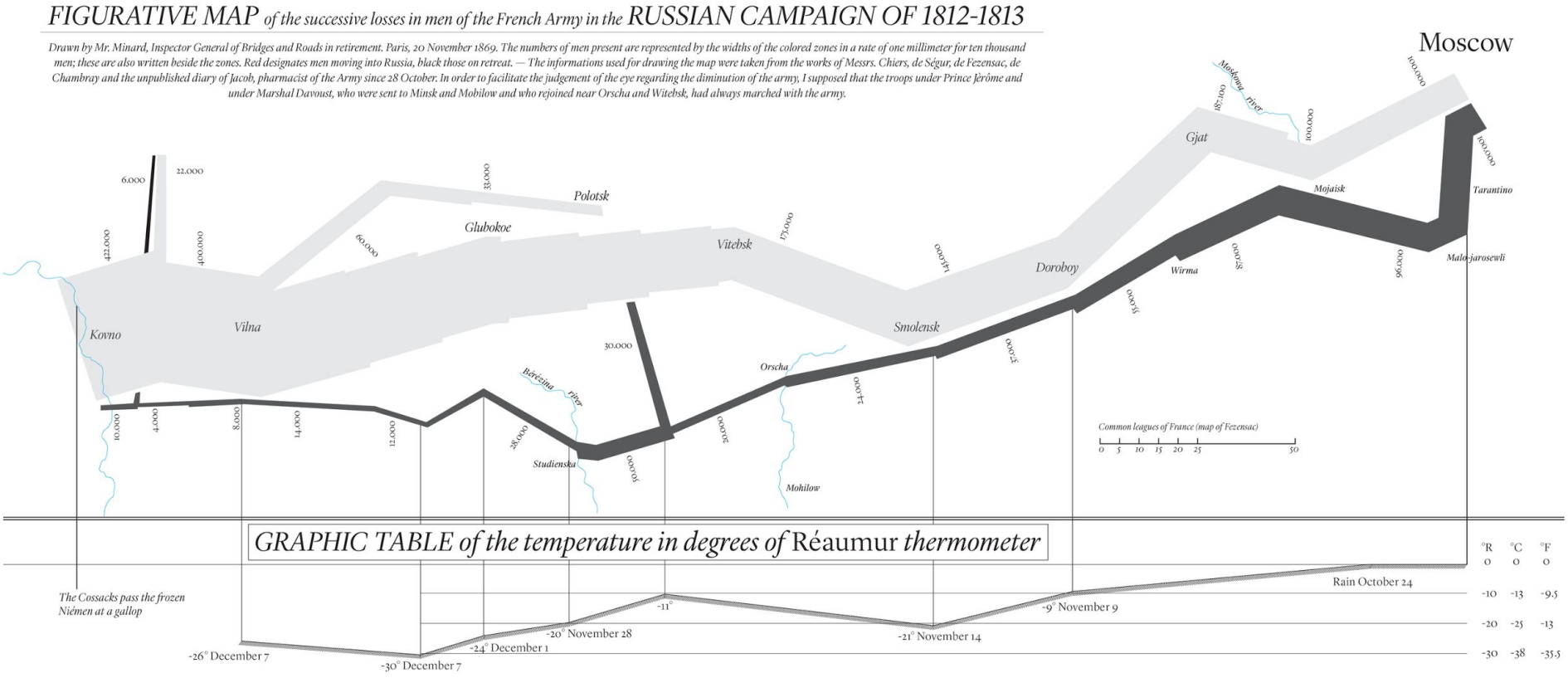
(Demo)

Minard's Map

Charles Joseph Minard, 1781-1870

- French civil engineer who created one of the greatest graphs of all time
- Visualized Napoleon's 1812 invasion of Russia, including
  - the number of soldiers
  - the direction of the march
  - the latitude and longitude of each city
  - the temperature on the return journey
  - Dates in November and December

Visualization of 1812 March



Different types of data

Longitude	Latitude	City	Direction	Survivors
32	54.8	Smolensk	Advance	145000
33.2	54.9	Dorogobouge	Advance	140000
34.4	55.5	Chjat	Advance	127100
37.6	55.8	Moscou	Advance	100000
34.3	55.2	Wixma	Retreat	55000
32	54.6	Smolensk	Retreat	24000
30.4	54.4	Orscha	Retreat	20000
26.8	54.3	Moiodexno	Retreat	12000

float:
decimal number

string:
text

int:
integer

(Demo)

Discussion Question

Longitude	Latitude	City	Direction	Survivors
32	54.8	Smolensk	Advance	145000
33.2	54.9	Dorogobouge	Advance	140000
34.4	55.5	Chjat	Advance	127100
37.6	55.8	Moscou	Advance	100000
34.3	55.2	Wixma	Retreat	55000
32	54.6	Smolensk	Retreat	24000
30.4	54.4	Orscha	Retreat	20000
26.8	54.3	Moiodexno	Retreat	12000

How would you calculate the average of the numbers in the last column?

A. `sum(minard.select('Survivors'))/minard.num_rows`
B. `sum(minard.column('Survivors'))/minard.num_rows`
C. Both **A** and **B** work.
D. Neither **A** nor **B** work.

Summary of Table Methods

- Creating and extending tables:
  - `Table().with_columns` and `Table.read_table`
- Finding the size: `num_rows` and `num_columns`
- Referring to columns: labels, relabeling, and indices
  - `labels` and `relabeled`; 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`, and so on
- Creating new tables containing some of the original columns:
  - `select`, `drop`

Practice

The table `students` has columns `Name`, `ID`, and `Score`.
For each part, write one line of code that evaluates to:

a) A table consisting of only the column labeled `Name`

b) The largest score

Practice

The table **`students`** has columns **`Name`**, **`ID`**, and **`Score`**.
For each part, write one line of code that evaluates to:

a) A table consisting of only the column labeled **`Name`**

```
students.select('Name')
```

```
students.select(0)
```

b) The largest score

```
students.column('Score').max()
```

```
max(students.column('Score'))
```

Sort

Sorting Tables

Tables are ordered collections of rows

- The `sort` method creates a new table with the same rows in a different order (the original table is unaffected)

- The `show` method displays the first rows of a table

(Demo)

Discussion Question

To create a table of the highest-paid players in each position:

```
nba.sort(3, descending=True).sort(1, distinct=True)
```

Which code creates a table of the lowest-paid players in each position?

A. `nba.sort(3, descending=True).sort(1, distinct=False)`
B. `nba.sort(3, descending=False).sort(1, distinct=True)`
C. `nba.sort(3, descending=False).sort(1, distinct=False)`
D. `nba.sort(3, descending=True).sort(1, distinct=True)`

Lists

Lists are Generic Sequences

A list is a sequence of values (just like an array), but the values can all have different types

```
[2+3, 'four', Table().with_column('K', [3, 4])]
```

If you create a table column from a list, it will be converted to an array automatically

(Demo)

Take

Take Rows, Select Columns

The `select` method returns a table with only some columns

The `take` method returns a table with only some rows

Take Rows, Select Columns

The `select` method returns a table with only some columns

The `take` method returns a table with only some rows

- Rows are numbered, starting at 0
- Taking a single number returns a one-row table
- Taking a list of numbers returns a table as well

(Demo)

Where

The Where Method

The `where` method specifies a column and a condition

It returns a new table with all rows satisfying the condition

(Demo)

Some Conditions

Predicate	Description
`are.equal_to(Z)`	Equal to `Z`
`are.above(x)`	Greater than `x`
`are.above_or_equal_to(x)`	Greater than or equal to `x`
`are.below(x)`	Less than `x`
`are.below_or_equal_to(x)`	Less than or equal to `x`
`are.between(x, y)`	Greater than or equal to `x` , and less than `y`
`are.strictly_between(x, y)`	Greater than `x` and less than `y`
`are.between_or_equal_to(x, y)`	Greater than or equal to `x` , and less than or equal to `y`
`are.containing(S)`	Contains the string `S`

You can also specify the negation of any of these conditions, by using `.not_` before the condition:

Predicate	Description
`are.not_equal_to(Z)`	Not equal to `Z`
`are.not_above(x)`	Not above `x`

Discussion Question

The table `nba` has columns `PLAYER`, `POSITION`, `TEAM`, `SALARY`.

Order the snippets of code to calculate the total salary of all small forwards (`SF`)

A. `nba` `.column(3)` `.sum()` `.where(1,'SF')`

B. `nba` `.where(1,'SF')` `.sum()` `.column(3)`

C. `nba` `.column(3)` `.where(1,'SF')` `.sum()`

D. `nba` `.where(1,'SF')` `.column(3)` `.sum()`

Discussion Question

The table **`nba`** has columns **`PLAYER`**, **`POSITION`**, **`TEAM`**, **`SALARY`**.

```
nba.where(1,'SF').column(3).sum()/nba.where(1,'SF').num_rows
```

What does this code compute?

(Demo)

Discussion Question

The table **`nba`** has columns **`PLAYER`**, **`POSITION`**, **`TEAM`**, **`SALARY`**.

Create an array containing the names of all point guards (**`PG`**) who make more than $15M/year

Discussion Question

The table **`nba`** has columns **`PLAYER`**, **`POSITION`**, **`TEAM`**, **`SALARY`**.

Create an array containing the names of all point guards (**`PG`**) who make more than $15M/year

```
nba.where(1, 'PG').where(3, are.above(15)).column(0)
```

(Demo)

Discussion Question

The table **`nba`** has columns **`PLAYER`**, **`POSITION`**, **`TEAM`**, **`SALARY`**.

What is the output when we execute a cell containing these two lines of code?

```
nba.with_row(['Jazz Bear','Mascot','Utah Jazz',100])
nba.where('PLAYER', are.containing('Bear'))
```

A. A table with one row for Jazz Bear
B. An empty table with no rows
C. An error message

(Demo)

Summary of Manipulating Rows

- `t.sort(column)`
  - sorts the rows in increasing order
- `t.take(row_numbers)`
  - keeps only specified rows (row numbers start at 0)
- `t.where(column, are.condition)`
  - keeps all rows for which a column's value satisfies a condition
- `t.where(column, value)`
  - keeps all rows containing a certain value in a column