COGS9-Intro to Data Science

Spring24 - Prof. Kyle Shannon

Discussion Section A01
Week 5

Teaching Assistant (TA): Abdullah
Instructional Assistant (IA): Kyra

Where to find all material

COGS 9

Search COGS 9 UCSD Podcast Gradescope

- Home
- Syllabus
- Readings
- Assignments
- Exam
- Final Project
- Office Hours
- Contact Us

This site uses Just the Docs, a

Introduction to Data Science

COGS 9 - UC San Diego - Prof. Kyle Shannon

Spring 2024 | SOLIS 107 | TU & TH 5:00-6:20PM

Welcome 👋

We are all very excited that you decided to join us on this whirlwind tour of data science. All relevant info, e.g. due dates, assignment links, etc. are found on this website. We look forward to teaching and working with all of you and hope to meet you in office hours. Check out the **Getting Started** section so you can hit the ground running when class starts!

NOTE
Week one I try to take as many students from the **waitlist** as I can, please email cogsadvising@ucsd.edu with further questions.

Discussion Sections

	Day	Time	Location	Staff	Materials
A01	Wed	12:00-12:50PM	CENTR 222	TA: Abdullah IAs: Kyra	View
A02	Wed	1:00-1:50PM	CENTR 222	TA: Kaushik IAs: Seshu, Vicky	View
A03	Wed	2:00-2:50PM	CENTR 222	TA: Matthew IAs: Jessica, Wenhua	View
A04	Wed	3:00-3:50PM	CENTR 222	TA: Vineeth IAs: Jiesen	View
A05	Wed	4:00-4:50PM	CENTR 222	TA: Vineeth IAs: Harshita	View

cogs9_TA Public

main | 1 Branch | 0 Tags | Go to file

AbdullahAshfaq Added week3 material

week2	Added week2 material
week3	Added week3 material
README.md	Update README.md

README

Cogs 9 Discussions-Intro to Data Science

Abdullah's discussion section material for COGS9 course

Upcoming Deadlines

Week 4		
Tue, Apr 23	LECT	Data Wrangling
Thu, Apr 25	LECT	Programming for Data Science
Fri, Apr 26	QUIZ	Reading Quiz 2 due
Fri, Apr 26	READ	Begin reading 3

Week 5		
Tue, Apr 30	LECT	Data Viz & Descriptive Analysis
Thu, May 02	LECT	Exploratory Data Analysis
Fri, May 03	QUIZ	Reading Quiz 3 due
Fri, May 03	READ	Begin reading 4

Week 6		
Mon, May 06	ASSG	Assignment 1 due
Tue, May 07	LECT	Communicating Data Science
Thu, May 09	LECT	Inferential Analysis

Week 7		
Mon, May 13	PROJ	Final Project Part 1 due

Discussion Sections Outline: Mostly Hands-on

- Week 2: Introductions, Making teams, Reading 1 (Part 1)
- Week 3: Reading 1 (Part 2), Python Basics with Jupyter Notebook
- Week 4: Reading 2, Getting data and wrangling it using Pandas
- Week 5: Reading 3, Assignment 1, Basics of SQL and Visualizations
- Week 6: Reading 4, Final Project Part 1 reviews/discussions
- Week 7: Assignment 2, Data Visualization and EDA demo
- Week 8: Assignment 3, Machine Learning demo
- Week 9: Reading 5, Closing thoughts
- Week 10: Final Project Part 2 reviews/discussions

Today’s Outline

Participation = Extra Credit 😀

- Reading 3 Summary
- Questions about Assignment 1
- Python Visualization
- Introduction to SQL

Reading 3
Tidy Data

Tidy Data

- Structure: Most statistical datasets are rectangular tables made up of *rows* and *columns*
- Definition: A dataset is a collection of values, usually either numbers (if quantitative) or strings (if qualitative)
- Values are organized in two different ways, Every value belongs to a *variable* and an *observation*

Example

Variable1	Variable2	Variable3
person	treatment	result
John Smith	a	—
Jane Doe	a	16
Mary Johnson	a	3
John Smith	b	2
Jane Doe	b	11
Mary Johnson	b	1

value value value

Variables in columns and observations in rows

Therefore, in tidy data:

- Each variable forms a column
- Each observation forms a row
- Each type of observational unit forms a table

The five most common issues with messy datasets

1. Columns headers are values, not variable names
2. Multiple variables are stored in one column
3. Variables are stored in both rows and columns
4. Multiple types of observational units are stored in the same table
5. A single observational unit is stored in multiple columns

File display

Column headers are values, not variable names

row	a	b	c
A	1	4	7
B	2	5	8
C	3	6	9

(a) Raw data

row	column	value
A	a	1
B	a	2
C	a	3
A	b	4
B	b	5
C	b	6
A	c	7
B	c	8
C	c	9

(b) Molten data

Table 5: A simple example of melting. (a) is melted with one colvar, row, yielding the molten dataset (b). The information in each table is exactly the same, just stored in a different way.

File display

Multiple variables stored in one column

country	year	column	cases
AD	2000	m014	0
AD	2000	m1524	0
AD	2000	m2534	1
AD	2000	m3544	0
AD	2000	m4554	0
AD	2000	m5564	0
AD	2000	m65	0
AE	2000	m014	2
AE	2000	m1524	4
AE	2000	m2534	4
AE	2000	m3544	6
AE	2000	m4554	5
AE	2000	m5564	12
AE	2000	m65	10
AE	2000	f014	3

(a) Molten data

country	year	sex	age	cases
AD	2000	m	0–14	0
AD	2000	m	15–24	0
AD	2000	m	25–34	1
AD	2000	m	35–44	0
AD	2000	m	45–54	0
AD	2000	m	55–64	0
AD	2000	m	65+	0
AE	2000	m	0–14	2
AE	2000	m	15–24	4
AE	2000	m	25–34	4
AE	2000	m	35–44	6
AE	2000	m	45–54	5
AE	2000	m	55–64	12
AE	2000	m	65+	10
AE	2000	f	0-14	3

(b) Tidy data

Table 10: Tidying the TB dataset requires first melting, and then splitting the `column` column into two variables: `sex` and `age`.

File display

Variables are stored in both rows and columns

id	date	element	value
MX17004	2010-01-30	tmax	27.8
MX17004	2010-01-30	tmin	14.5
MX17004	2010-02-02	tmax	27.3
MX17004	2010-02-02	tmin	14.4
MX17004	2010-02-03	tmax	24.1
MX17004	2010-02-03	tmin	14.4
MX17004	2010-02-11	tmax	29.7
MX17004	2010-02-11	tmin	13.4
MX17004	2010-02-23	tmax	29.9
MX17004	2010-02-23	tmin	10.7

(a) Molten data

id	date	tmax	tmin
MX17004	2010-01-30	27.8	14.5
MX17004	2010-02-02	27.3	14.4
MX17004	2010-02-03	24.1	14.4
MX17004	2010-02-11	29.7	13.4
MX17004	2010-02-23	29.9	10.7
MX17004	2010-03-05	32.1	14.2
MX17004	2010-03-10	34.5	16.8
MX17004	2010-03-16	31.1	17.6
MX17004	2010-04-27	36.3	16.7
MX17004	2010-05-27	33.2	18.2

(b) Tidy data

Multiple types in one table

File display

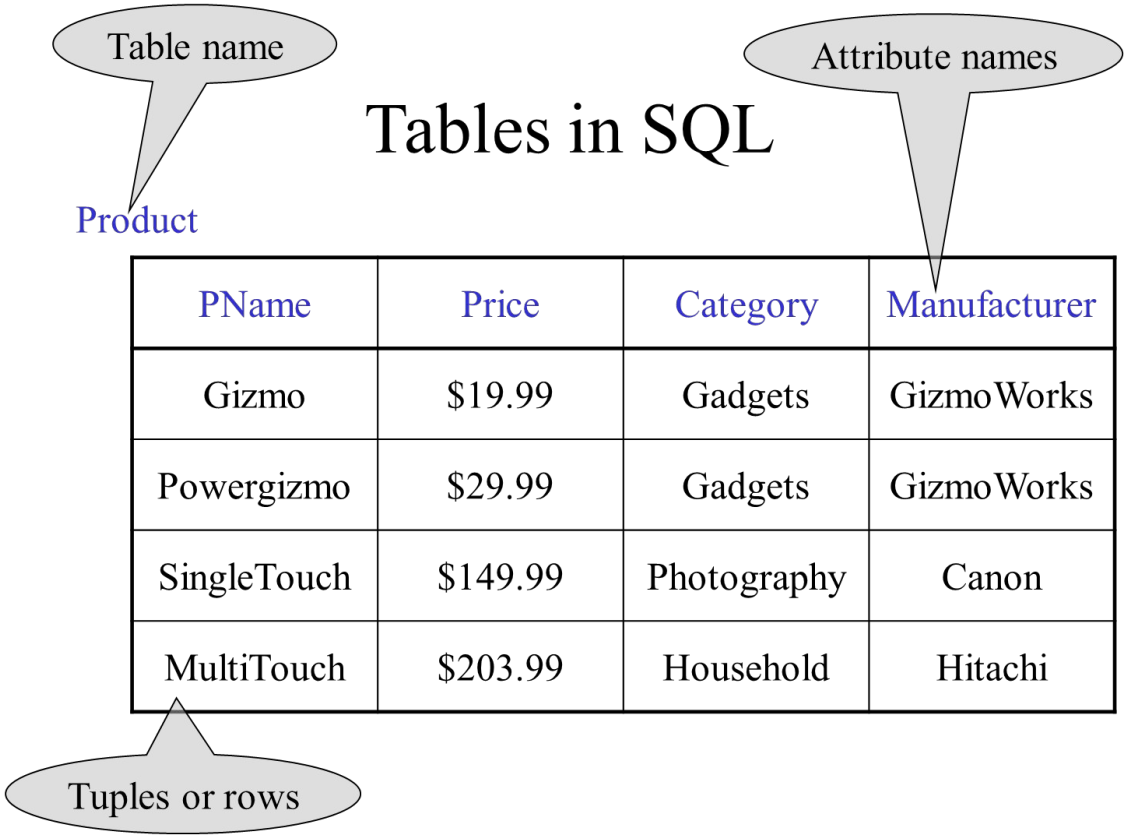
year	artist	time	track	date	week	rank
2000	2 Pac	4:22	Baby Don't Cry	2000-02-26	1	87
2000	2 Pac	4:22	Baby Don't Cry	2000-03-04	2	82
2000	2 Pac	4:22	Baby Don't Cry	2000-03-11	3	72
2000	2 Pac	4:22	Baby Don't Cry	2000-03-18	4	77
2000	2 Pac	4:22	Baby Don't Cry	2000-03-25	5	87
2000	2 Pac	4:22	Baby Don't Cry	2000-04-01	6	94
2000	2 Pac	4:22	Baby Don't Cry	2000-04-08	7	99
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-02	1	91
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-09	2	87
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-16	3	92
2000	3 Doors Down	3:53	Kryptonite	2000-04-08	1	81
2000	3 Doors Down	3:53	Kryptonite	2000-04-15	2	70
2000	3 Doors Down	3:53	Kryptonite	2000-04-22	3	68
2000	3 Doors Down	3:53	Kryptonite	2000-04-29	4	67
2000	3 Doors Down	3:53	Kryptonite	2000-05-06	5	66

Visualizations in Python

Overview of SQL

Credits: https://courses.cs.washington.edu/courses/cse544/06sp/lectures/lecture-sql.ppt

Standard language for querying and manipulating data (**S**tructured **Q**uery **L**anguage)



Data Types in SQL

•Atomic types:

–Characters: CHAR(20), VARCHAR(50)

–Numbers: INT, BIGINT, SMALLINT, FLOAT

–Others: DATETIME, …

SQL Query

Basic form: (plus many many more bells and whistles)

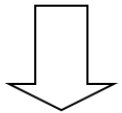
```
SELECT  <attributes>
FROM    <one or more relations>
WHERE   <conditions>
```

Simple SQL Query

Product

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

```
SELECT  *
FROM    Product
WHERE   category='Gadgets'
```



PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks

"selection"

Aggregation

```
SELECT  avg(price)
FROM    Product
WHERE   maker="Toyota"
```

```
SELECT  count(*)
FROM    Product
WHERE   year > 1995
```

SQL supports several aggregation operations:

sum, count, min, max, avg

Except count, all aggregations apply to a single attribute

Grouping and Aggregation

Purchase(product, date, price, quantity)

Find total sales after 10/1/2005 per product.

```
SELECT    product, Sum(price*quantity) AS TotalSales
FROM      Purchase
WHERE     date > '10/1/2005'
GROUP BY  product
```

Let's see what this means…

Filtering (Python vs SQL)

```
df1 = csv_data[csv_data['housing_median_age']<20]
df1
```

	longitude	latitude	housing_median_age	total_rooms
4	-119.67	36.33	19	1241
7	-120.65	35.48	19	2310
8	-122.84	38.40	15	3080
13	-117.03	32.97	16	3936

Select * from csv_data where “housing_media_age”<20;

Sorting (Python vs SQL)

```
csv_data.sort_values('total_rooms', ascending=True)
```

	longitude	latitude	housing_median_age	total_rooms	total
1115	-116.95	33.86	1	6	
740	-117.12	32.66	52	16	
2640	-114.62	33.62	26	18	
641	-121.04	37.67	16	19	
2690	-118.06	34.03	36	21	

Select * from csv_data
order by “total_rooms” ASC;

Aggregating (Python vs SQL)

```
df3 = df.groupby(['Sex','Year']).agg(
    avg_sleep=('Avg hrs per day sleeping','mean')
).reset_index()

df3.head()
```

	Sex	Year	avg_sleep
0	Both	2003	8.702857
1	Both	2004	8.678571
2	Both	2005	8.745714
3	Both	2006	8.757143
4	Both	2007	8.697619

```
select
"Sex", "Year",
avg("Avg_hrs_per_day_sleeping"::float) as
avg_sleep
from df
group by 1,2;
```

SQL Resources

Online tool to practice: https://sqliteonline.com/

Cheatsheet: https://images.datacamp.com/image/upload/v1714149594/Marketing/Blog/SQL_for_Data_Science.pdf

Tutorials: https://www.mysqltutorial.org/mysql-basics/