A COURSE ON DATA WRANGLING AND VISUALIZATION

Nelson Uhan

Data Science Seminar
11 February 2021

- Last semester, Jay Foraker and I developed a new course on data wrangling and visualization for operations research majors

- What is this course about?

- Structured ways to think about data wrangling and visualization

MOTIVATION FOR THE COURSE

- The operations research (OR) curriculum exposes students to a wide variety of modeling and algorithmic techniques
- These concepts have been typically taught with small, tidy data sets
- Unfortunately, as a result, students have not been well-equipped to tackle the large, messy data sets typically involved with OR capstone projects
- This course aims to fill this gap

STUDENT GOALS FOR THE COURSE

1. Learn to create useful visualizations of data through a **grammar of graphics**

2. Learn to wrangle (i.e. clean and manipulate) large, messy data sets into forms suitable for modeling and analysis through a **grammar of data manipulation**

3. Increase general fluency with Python

GRAMMAR OF GRAPHICS

- A **grammar of graphics** allows us to concisely describe the components of a (statistical) visualization
- With such a grammar, we can move beyond named graphics (e.g. "scatterplot", "bar graph") and specify basic and complex visualizations in a structured way
- Popularized by Hadley Wickham (Chief Scientist, R Studio) and his ggplot2 package for R
- In this course, we used Altair, a grammar-based visualization package for Python

COMPONENTS OF A VISUALIZATION

Note. This is Altair's terminology.

- **Dataset** with variables and observations
- **Encoding channels** map variables to visual attributes (e.g. x-position, y-position, color, shape)
    - **Scales** for each encoding channel to adjust these mappings (e.g. linear vs. log scale positions)
- **Graphical marks** specify how these visual attributes should be visually represented (e.g. points, lines, bars)
- **Transformations** modify the data before visualization
- **Layering**, **concatenation**, and **faceting** specify how to combine or generate multiple related charts

AN EXAMPLE

- Let's use some health and population data for a number of countries between 1955 and 2005
    - Data from by the Gapminder Foundation

- First, let's take a look at the first 5 rows of the data:

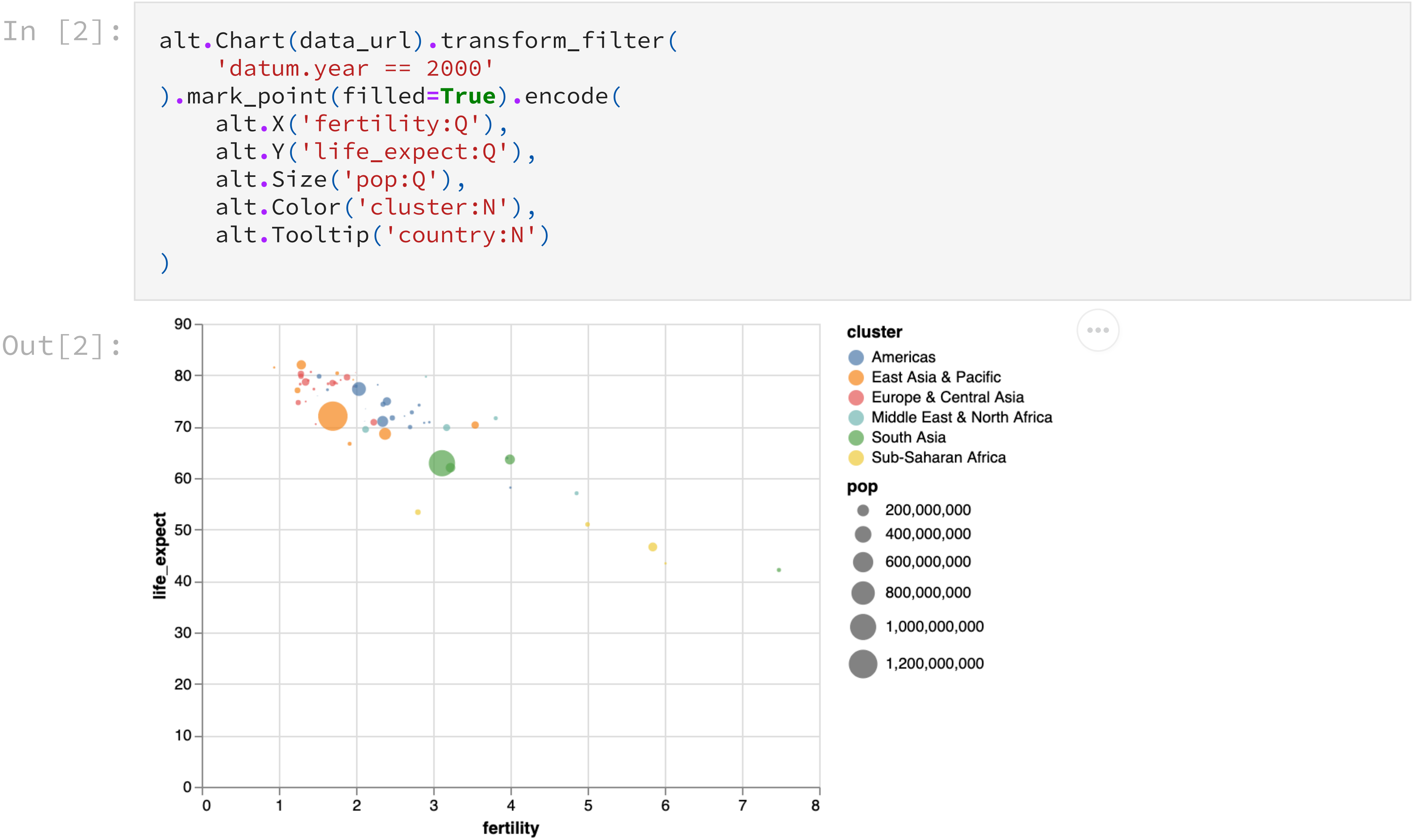
	year	country	cluster	pop	life_expect	fertility
0	1955	Afghanistan	South Asia	8891209	30.332	7.7
1	1960	Afghanistan	South Asia	9829450	31.997	7.7
2	1965	Afghanistan	South Asia	10997885	34.020	7.7
3	1970	Afghanistan	South Asia	12430623	36.088	7.7
4	1975	Afghanistan	South Asia	14132019	38.438	7.7

- Each row of this dataset contains the following data for each `country` and `year`:
    - region of the world (`cluster`)
    - total population (`pop`)
    - average life expectancy in years (`life_expect`)
    - number of children per woman (`fertility`)

- Let's use Altair to plot the average life expectancy (`life_expect`) vs. number of children per woman (`fertility`) in the year 2000

In [1]:

```
import altair as alt

data_url = 'data/gapminder.csv'
```



In [2]:

```
alt.Chart(data_url).transform_filter(
    'datum.year == 2000'
).mark_point(filled=True).encode(
    alt.X('fertility:Q'),
    alt.Y('life_expect:Q'),
    alt.Size('pop:Q'),
    alt.Color('cluster:N'),
    alt.Tooltip('country:N')
)
```

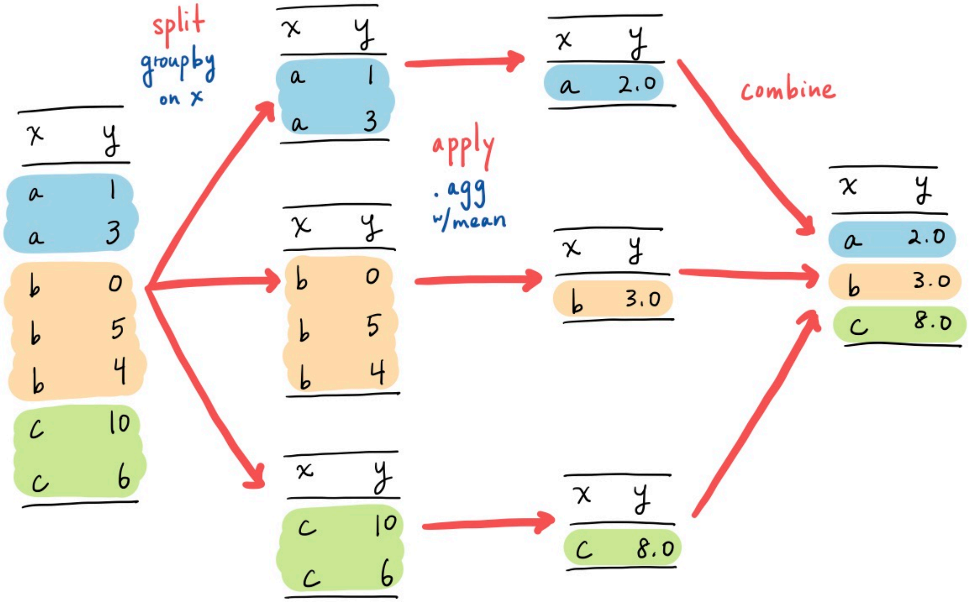
Out[2]:

GRAMMAR OF DATA MANIPULATION

- A **grammar of data manipulation** allows us to use a consistent set of operations to solve the most common data manipulation challenges
- dplyr (also by Hadley Wickham) is a R package that embodies this notion
    - One function per operation
    - Ideas from relational databases, query languages (e.g. SQL)
- In this course, we used Pandas, the popular Python data manipulation library
- Unfortunately, Pandas is notorious for having multiple ways of doing the same operation
- For this course, we restricted ourselves to an *opinionated subset* of Pandas, with one way to achieve each operation

BASIC OPERATIONS FOR DATA MANIPULATION

- **Filter rows** based on their values
- **Select and drop columns** based on their names
- **Sort rows** based on their values
- **Create new columns** that are functions of existing columns
- Aggregate, transform, and filter **groups of data** through **split-apply-combine**
- **Pivot** data from long form to wide form and vice versa
- **Merge** datasets together based on key columns


split
groupby
on x
apply
.agg
w/mean
combine
x y
a 1
a 3
b 0
b 5
b 4
c 10
c 6
x y
a 1
a 3
x y
b 0
b 5
b 4
x y
c 10
c 6
x y
a 2.0
x y
b 3.0
x y
c 8.0
x y
a 2.0
b 3.0
c 8.0

AN EXAMPLE

- Let's read the data into a Pandas DataFrame:

In [4]:

```
import pandas as pd

df = pd.read_csv('data/gapminder.csv')
df.head()
```

Out[4]:

	year	country	cluster	pop	life_expect	fertility
0	1955	Afghanistan	South Asia	8891209	30.332	7.7
1	1960	Afghanistan	South Asia	9829450	31.997	7.7
2	1965	Afghanistan	South Asia	10997885	34.020	7.7
3	1970	Afghanistan	South Asia	12430623	36.088	7.7
4	1975	Afghanistan	South Asia	14132019	38.438	7.7

- Let's find the country with the highest life-expectancy-to-fertility ratio in each year between 1980 to 2000 in the dataset.

In [5]:

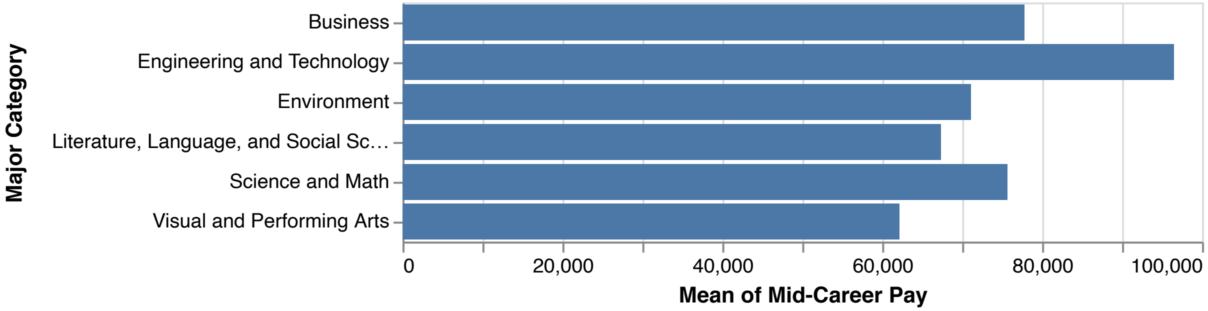
```
(
    df
    .assign(
        ratio=lambda x: x['life_expect'] / x['fertility']
    )
    .sort_values(['year', 'ratio'], ascending=[True, False])
    .groupby('year')
    .agg(
        highest_ratio_country=('country', 'first'),
        highest_ratio=('ratio', 'first')
    )
    .reset_index()
    .query('year >= 1980 and year <= 2000')

)
```

Out[5]:

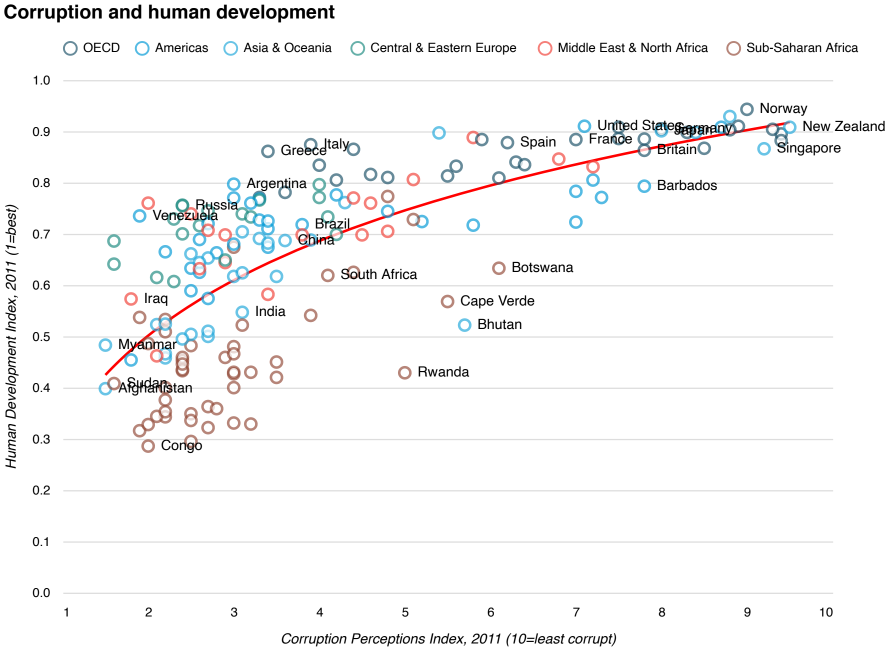
	year	highest_ratio_country	highest_ratio
5	1980	Germany	50.547945
6	1985	Hong Kong	58.167939
7	1990	Spain	61.078740
8	1995	Hong Kong	74.074074
9	2000	Hong Kong	86.696809

COURSE PROJECTS

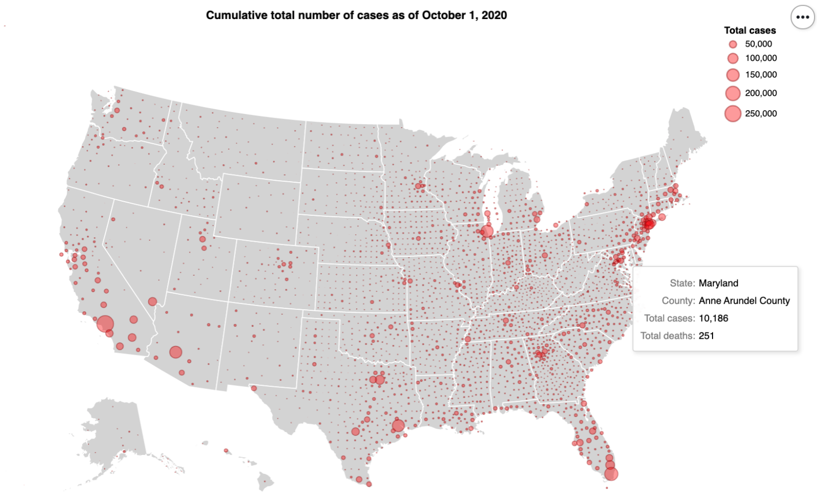
Project 1. Visually explore the relationships between colleges, majors, and salary after graduation, using data from PayScale



Project 2. Practice layering and customizing Altair charts by reproducing this graphic showing the relationship between corruption and human development, originally published in *The Economist* in 2011.



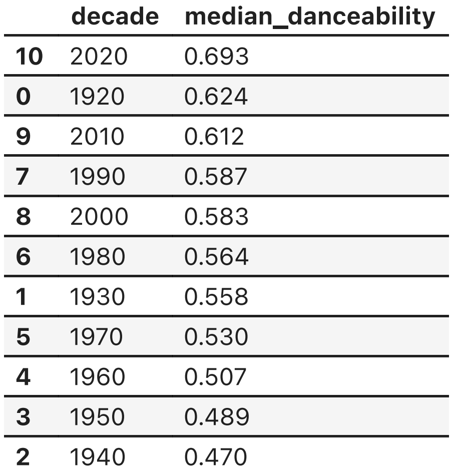
Project 3. Create several visualizations of COVID-19 in the United States, using a dataset created by the instructors, based on COVID-19 cases and deaths data from usafacts.org, combined with geographic data from the United States Census Bureau.



State: Maryland
County: Anne Arundel County
Total cases: 10,186
Total deaths: 251

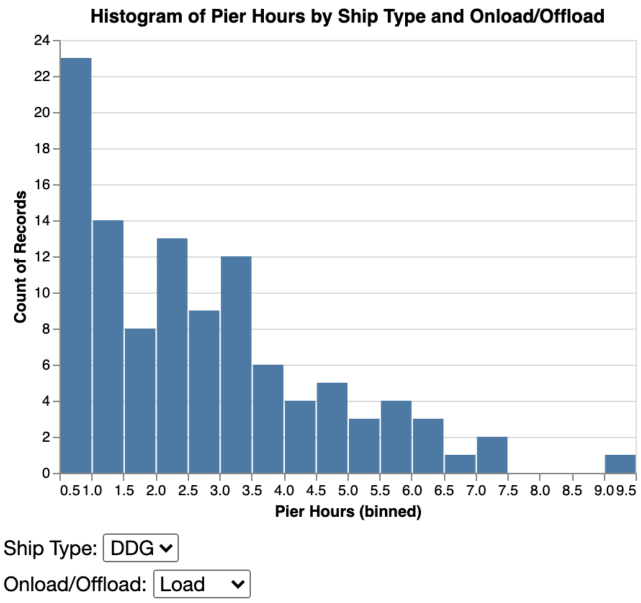
Project 5. Use Pandas to recreate the dataset used in Project 3.

Project 4. Use Pandas to explore this Kaggle dataset containing the following information on over 160,000 tracks on Spotify, including measures such as *acousticness*, *danceability*, *energy*, *instrumentalness*, *liveness*, and *speechiness*.



	decade	median_danceability
10	2020	0.693
0	1920	0.624
9	2010	0.612
7	1990	0.587
8	2000	0.583
6	1980	0.564
1	1930	0.558
5	1970	0.530
4	1960	0.507
3	1950	0.489
2	1940	0.470

Project 6. Work with a (perturbed) dataset on ordnance onload and offload operations performed by Navy Munitions Command Atlantic (NMCLANT) Detachment (Det) Sewells Point in 2015. Clean the data, compute performance metrics, create interactive visualizations of the data.



Ship Type: DDG

Onload/Offload: Load

OTHER TOPICS WE HAD HOPED TO COVER

- Getting data through **web scraping**
- Getting data through **website APIs**
- Interoperability between R and Python
    - rpy2 is a Python library that lets you call R directly from inside Python
    - reticulate is an R library that lets you do the opposite: call Python directly from inside R

IF YOU'RE INTERESTED...

- Data visualization with Altair:

  J. Heer, D. Moritz, J. VanderPlas, B. Craft. *University of Washington Data Visualization Curriculum*. Set of Jupyter notebooks. [link]

- Data visualization and wrangling concepts, taught through R:

  H. Wickham, G. Grolemund. *R for Data Science*. Electronic book, physical copy published by O'Reilly, 2017. [link]

- Altair documentation [link]

- Pandas documentation [link]

- Course website with materials:

  https://www.usna.edu/Users/math/uhan/sa463a/