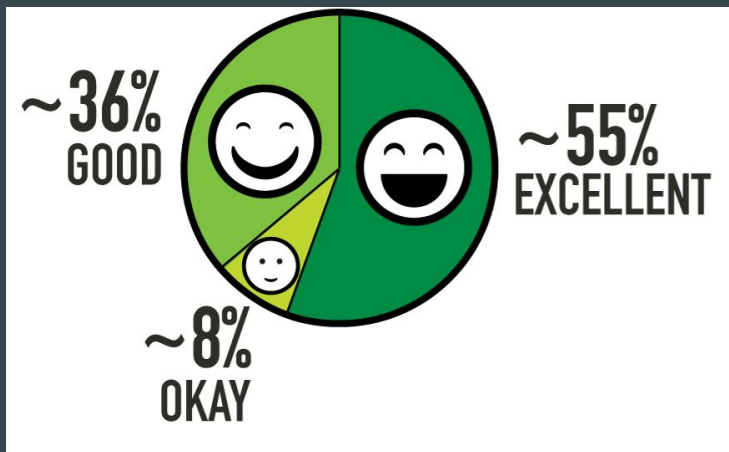# Seattle Food Inspections

•••

David Clancy, Sadra Fardhosseini, Greg Talpey, Harry Xie

Introduction

- Starting 2012, jurisdictions across the country including King County have begun publishing **health inspection scores** using a standardized scoring system called LIVES.
- This open data allowed restaurant consumers to make informed decisions based on **where they want to eat** and motivated a lot of restaurant establishments to **improve their inspection score** in the hopes of attracting a bigger customer base.

Brief overview of the objectives

1. **Location Specific Contributions**: Where are the specific regions in Seattle that have the most high risk food safety violations?
2. **Interactive Map**: showing food inspection ratings in Seattle.
3. Then overlay these **ratings with demographic** information in the areas around restaurants.
4. Users will be able to explore **correlations** between the demographics and health inspection ratings to help make decisions about where to eat.

Package Dependency
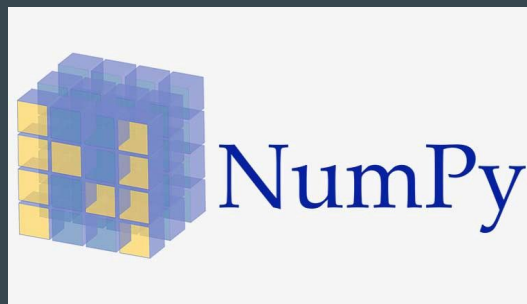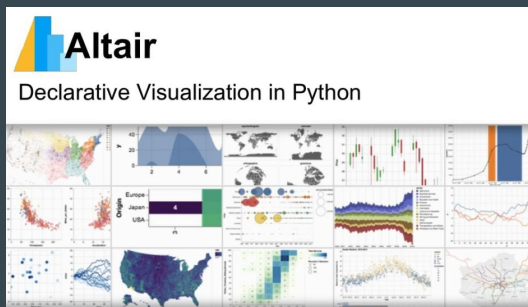
Dataset

1. Food Establishment Inspection Data - *King County Open Data*

	Name	Program Identifier	Inspection Date	Description	Address	City	Zip Code	Phone	Longitude	Latitude	Inspection Business Name	Inspection Type
0	@ THE SHACK, LLC	SHACK COFFEE	08/03/2019	Seating 0-12 - Risk Category III	2920 SW AVALON WAY	Seattle	98126	(206) 938-5665	-122.370913	47.570425	@ THE SHACK, LLC	Routine Inspection/Field Review

2. Median Income 5-Year Estimates (S1903) - *American Community Survey*
3. Marital Status (S1201) - *American Community Survey*

	zipcode	Population	No_Married(%)	Widowed(%)	Divorced(%)	Separated(%)	Never_Married(%)	No_Married	Widowed	Divorced	Separated	Never_Married
0	83501	28808	52.0	6.7	13.5	1.1	26.7	14980	1930	3889	316	7691
1	83822	1625	52.1	8.2	14.2	2.0	23.5	846	133	230	32	381

Data Cleaning

- Detecting **missing values** (tolerance)
  - (✅)Missing Violation
  - (✕)Missing Name / Grade
  - (✕)Unnecessary Identifiers
- Extracting desired **rows, columns**
- Modifying the data (i.e. **type**, caps, etc.)
- Joining the census data by **zip codes**

```
SEATTLE           136455
BELLEVUE           19689
KENT               13705
RENTON             11805
FEDERAL WAY        11695
REDMOND            11159
KIRKLAND           10654
AUBURN              7752
ISSAQUAH            5846
SHORELINE           5185
Name: City, dtype: int64
```

Data Cleaning

- Shape: **(274970, 22) >>> (8413, 12)**
- Columns & Types

```
RangeIndex: 274970 entries, 0 to 274969
Data columns (total 22 columns):
Name                        274970 non-null object
Program Identifier          274970 non-null object
Inspection Date             274423 non-null object
Description                 274970 non-null object
Address                     274970 non-null object
City                        274970 non-null object
Zip Code                    274970 non-null object
Phone                       192911 non-null object
Longitude                   274959 non-null float64
Latitude                    274959 non-null float64
Inspection Business Name    274423 non-null object
Inspection Type             274423 non-null object
Inspection Score            274369 non-null float64
Inspection Result           274423 non-null object
Inspection Closed Business  274423 non-null object
Violation Type              156246 non-null object
Violation Description       156246 non-null object
Violation Points            274970 non-null int64
Business_ID                 274970 non-null object
Inspection_Serial_Num       274423 non-null object
Violation_Record_ID         156246 non-null object
Grade                       215640 non-null float64
dtypes: float64(4), int64(1), object(17)
```

```
Int64Index: 8413 entries, 134890 to 208625
Data columns (total 12 columns):
Name                        8413 non-null object
Inspection Date             8413 non-null datetime64[ns]
Description                 8413 non-null object
Zip Code                    8413 non-null int64
Inspection Type             8413 non-null object
Inspection Score            8413 non-null float64
Inspection Result           8413 non-null object
Inspection Closed Business  8413 non-null object
Violation Type              3853 non-null object
Violation Description       3853 non-null object
Violation Points            8413 non-null int64
Grade                       8413 non-null float64
dtypes: datetime64[ns](1), float64(2), int64(2), object(7)
```

Use Case

- At least in NW Ethiopia, **certain demographic information** of the workers is associated with food handling practices.
- Users can see visually if such **correlation** between restaurant locations and some of the same demographic information (median income and various marital statuses) in Seattle.
- Users can use these **demographic statistics** to find what zip codes in Seattle to explore for new restaurants.

Demo

(transition to JupyterLab for demonstration)

Design

Data collection from URLs (inspections, census and geographic)

⬇

Cleaning datasets (remove null/NaN etc, erroneous locations ...)
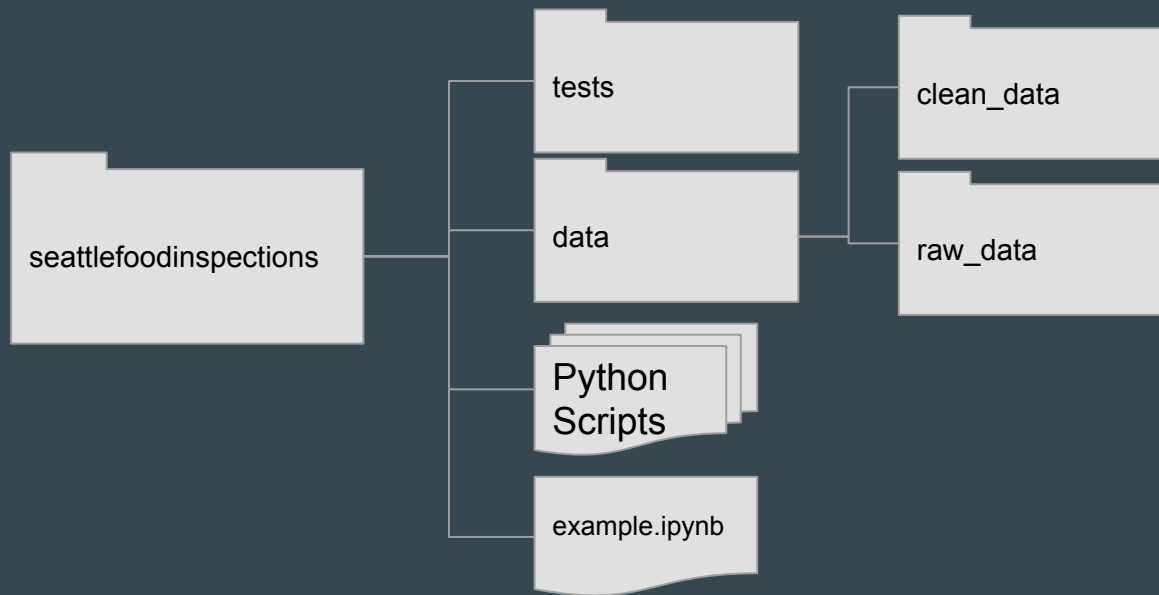
⬇

Merging datasets (merge on restaurant zip code)

⬇

Visualizing (chloropleth of census data with interactive histograms, associated trend analysis charts)

(Partial) File Structure

seattlefoodinspections

tests

data

Python
Scripts

example.ipynb

clean_data

raw_data

Lessons Learned & Future Work

- Data sets are inherently "dirty" -> data science software requires significant data cleaning and preparation
- Modulation and generalization has up-front costs that pay for themselves in the long run
  - It's easy to write functions that work for one instance but which break easily

- Incorporate more robust census data
- Generalize functions to be used for any city/region

Questions?

Thank you