

7.1) Data Cleaning

Vitor Kamada

December 2019

Tables, Graphics, and Figures from

Principles and Techniques of Data Science

Lau et al. (2019): Ch 5 Data Cleaning

https://www.textbook.ds100.org/ch/05/cleaning_intro.html

Berkeley Police Department - Calls for Service

```
import numpy as np
import pandas as pd
path = 'https://github.com/DS-100/textbook/raw/master/content/'
calls = pd.read_csv(path + 'ch/05/data/Berkeley_PD_-_Calls_for_Service.csv')
calls.head(3)
```

	CASENO	OFFENSE	EVENTDT	EVENTTM	CVLEGEND
0	17091420	BURGLARY AUTO	07/23/2017 12:00:00 AM	06:00	BURGLARY - VEHICLE
1	17020462	THEFT FROM PERSON	04/13/2017 12:00:00 AM	08:45	LARCENY
2	17050275	BURGLARY AUTO	08/24/2017 12:00:00 AM	18:30	BURGLARY - VEHICLE

Missing Values

```
null_rows = calls.isnull().any(axis=1)
calls[null_rows]
```

Block_Location	BLKADDR	City	State
Berkeley, CA\n(37.869058, -122.270455)	NaN	Berkeley	CA
Berkeley, CA\n(37.869058, -122.270455)	NaN	Berkeley	CA

27 calls didn't have a recorded address in BLKADDR

```
def combine_event_datetimes(calls):
    combined = pd.to_datetime(
        # Combine date and time strings
        calls['EVENTDT'].str[:10] + ' ' + calls['EVENTTM'],
        infer_datetime_format=True,)
    return calls.assign(EVENTDTTM=combined)

# To peek at the result without mutating the calls DF:
calls.pipe(combine_event_datetimes)[['EVENTDT',
                                     'EVENTTM', 'EVENTDTTM']]
```

	EVENTDT	EVENTTM	EVENTDTTM
0	07/23/2017 12:00:00 AM	06:00	2017-07-23 06:00:00
1	04/13/2017 12:00:00 AM	08:45	2017-04-13 08:45:00
2	08/24/2017 12:00:00 AM	18:30	2017-08-24 18:30:00

```
calls['OFFENSE'].unique()
```

```
['BURGLARY AUTO', 'THEFT FROM PERSON', 'GUN/WEAPON',  
'VEHICLE STOLEN', 'BURGLARY RESIDENTIAL', 'VANDALISM',  
'DISTURBANCE', 'THEFT MISD. (UNDER $950)', 'THEFT FROM AUTO',  
'DOMESTIC VIOLENCE', 'THEFT FELONY (OVER $950)', 'ALCOHOL OFFENSE',  
'MISSING JUVENILE', 'ROBBERY', 'IDENTITY THEFT',  
'ASSAULT/BATTERY MISD.', '2ND RESPONSE', 'BRANDISHING',  
'MISSING ADULT', 'NARCOTICS', 'FRAUD/FORGERY',  
'ASSAULT/BATTERY FEL.', 'BURGLARY COMMERCIAL', 'MUNICIPAL CODE',  
'ARSON', 'SEXUAL ASSAULT FEL.', 'VEHICLE RECOVERED',  
'SEXUAL ASSAULT MISD.', 'KIDNAPPING', 'VICE', 'HOMICIDE'],
```

```
calls['CVLEGEND'].unique()
```

```
'BURGLARY - VEHICLE', 'LARCENY', 'WEAPONS OFFENSE',  
'MOTOR VEHICLE THEFT', 'BURGLARY - RESIDENTIAL', 'VANDALISM',  
'DISORDERLY CONDUCT', 'LARCENY - FROM VEHICLE', 'FAMILY OFFENSE',  
'LIQUOR LAW VIOLATION', 'MISSING PERSON', 'ROBBERY', 'FRAUD',  
'ASSAULT', 'NOISE VIOLATION', 'DRUG VIOLATION',  
'BURGLARY - COMMERCIAL', 'ALL OTHER OFFENSES', 'ARSON',  
'SEX CRIME', 'RECOVERED VEHICLE', 'KIDNAPPING', 'HOMICIDE'],
```

Inconsistencies in the BLKADDR column

```
calls['BLKADDR'][[0, 5001]]
```

0	2500 LE CONTE AVE
5001	ALLSTON WAY & FIFTH ST

Address is recorded but other times a cross street is recorded

```
def split_lat_lon(calls):
    return calls.join(
        calls['Block_Location']
        # Get coords from string
        .str.split('\n').str[2]
        # Remove parens from coords
        .str[1:-1]
        # Split latitude and longitude
        .str.split(', ', expand=True)
        .rename(columns={0: 'Latitude', 1: 'Longitude'}))

calls.pipe(split_lat_lon).head(2)
```

Block_Location	BLKADDR	City	State	Latitude	Longitude
2500 LE CONTE AVE\nBerkeley, CA\n(37.876965, -...	2500 LE CONTE AVE	Berkeley	CA	37.876965	-122.260544
2200 SHATTUCK AVE\nBerkeley, CA\n(37.869363,	2200 SHATTUCK	Berkeley	CA	37.869363	-122.268028


```
day_of_week = pd.read_csv(path + 'ch/05/data/cvdow.csv')
```

	CVDOW	Day
0	0	Sunday
1	1	Monday
2	2	Tuesday
3	3	Wednesday
4	4	Thursday
5	5	Friday
6	6	Saturday

```
def match_weekday(calls):
    return calls.merge(day_of_week, on='CVDOW')
calls.pipe(match_weekday).head(2)
```

CVDOW	InDbDate	Block_Location	BLKADDR	City	State	Day
0	08/29/2017 08:28:05 AM	2500 LE CONTE AVE\nBerkeley, CA\n(37.876965, -...	2500 LE CONTE AVE	Berkeley	CA	Sunday
0	08/29/2017 08:28:03 AM	BOWDITCH STREET & CHANNING WAY\nBerkeley,	BOWDITCH STREET & CHANNING WAY	Berkeley	CA	Sunday

```
def drop_unneeded_cols(calls):
    return calls.drop(columns=['CVDOW', 'InDbDate', 'Block_Location',
                              'City', 'State', 'EVENTDT', 'EVENTTM'])
calls_final = (calls.pipe(combine_event_datetimes).pipe(split_lat_lon)
               .pipe(match_weekday).pipe(drop_unneeded_cols))
df_interact(calls_final)
```

row 0

col 0

	CASENO	OFFENSE	CVLEGEND
0	17091420	BURGLARY AUTO	BURGLARY - VEHICLE
1	17038302	BURGLARY AUTO	BURGLARY - VEHICLE

```
calls_final['EVENTDTTM'].dt.date.sort_values()
```

```
1277    2017-03-02
      ...
256     2017-08-27
```

```
(calls_final['EVENTDTTM'].dt.date.max() -  
 calls_final['EVENTDTTM'].dt.date.min())
```

```
datetime.timedelta(179)
```

The table contains data for a time period of 179 days which is close enough to the 180 day time period in the data description

```
import folium # Use the Folium Javascript Map Library
import folium.plugins
SF_COORDINATES = (37.87, -122.28)
sf_map = folium.Map(location=SF_COORDINATES, zoom_start=13)
locs = calls_final[['Latitude',
                    'Longitude']].dropna().astype('float').as_matrix()
heatmap = folium.plugins.HeatMap(locs.tolist(), radius = 10)
sf_map.add_child(heatmap)
```



```
import seaborn as sns
sns.set()
sns.set_context('talk')
calls_final['CASENO'].plot.hist(bins=30)
```

