Privacy

David Wagner

What *is* privacy?

“the right to be let alone”

— Justices Samuel Warren
and Louis Brandeis

control over who can obtain or use information about me

Fair Information Practices

notice, consent (opt-in/opt-out), access

contextual integrity

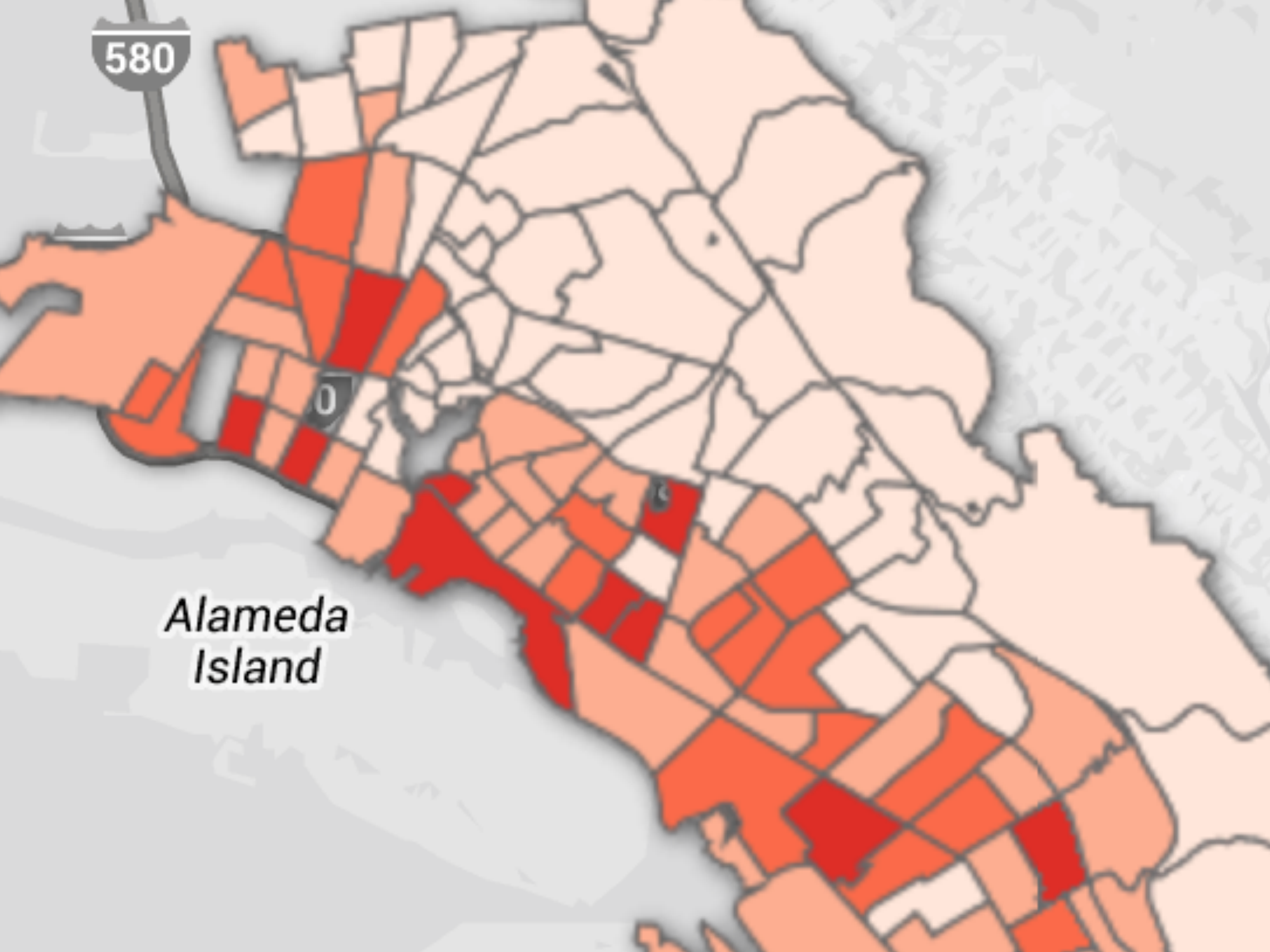
— Helen Nissenbaum

This class:
informational privacy.

What does someone else know about me?
What can they infer, from what they know?
What choices do I have?

A worked example:

Automated license plate readers


580
0
Alameda
Island

Ashby Ave
Alcatraz Ave
Shattuck Ave
Hollis St
Emeryville
51st St
13
Round Top
Moraga Way
Ivy Dr
Pinehurst Rd
Canyon
Piedmont
80
24
Maritime St
Adeline St
27th St
Grand Ave
Mandana Blvd
Park Blvd
Redwood Regional Park
Redwood Rd
7th St
Peralta St
14th St
Oakland
880
8th St
International Blvd
14th Ave
23rd Ave
35th Ave
38th Ave
Campus Dr
Keller Ave
Main St
61
Lincoln Ave
Encinal Ave
Alameda
Foothill Blvd
Mountain Blvd
Oak Port St
High St
Otis Dr
Shoreline Dr
Alameda Harbor Bay - San Francisco Ferry Building
185
80th Ave
82nd Ave
MacArthur Blvd
90th Ave
98th Ave
NIMITZ FWY
San Leandro

License plates

We're going to look at some data collected by the Oakland Police Department. They have automated license plate readers on their police cars, and they've built up a database of license plates that they've seen -- and where and when they saw each one.

Data collection

First, we'll gather the data. It turns out the data is publicly available on the Oakland public records site. I downloaded it and combined it into a single CSV file by myself before lecture.

```
lprs = Table.read_table('./all-lprs.csv.gz', compression='gzip', sep=',')
```

```
lprs.column_labels
```

```
('red_VRM', 'red_Timestamp', 'Location')
```

Let's start by renaming some columns, and then take a look at it.

```
lprs.relabel('red_VRM', 'Plate')
lprs.relabel('red_Timestamp', 'Timestamp')
lprs
```

Plate	Timestamp	Location
1275226	01/19/2011 02:06:00 AM	(37.798304999999999, -122.27574799999999)
27529C	01/19/2011 02:06:00 AM	(37.798304999999999, -122.27574799999999)
1158423	01/19/2011 02:06:00 AM	(37.798304999999999, -122.27574799999999)
1273718	01/19/2011 02:06:00 AM	(37.798304999999999, -122.27574799999999)
1077682	01/19/2011 02:06:00 AM	(37.798304999999999, -122.27574799999999)
1214195	01/19/2011 02:06:00 AM	(37.798281000000003, -122.27575299999999)
1062420	01/19/2011 02:06:00 AM	(37.79833, -122.27574300000001)
1319726	01/19/2011 02:05:00 AM	(37.798475000000003, -122.27571500000001)
1214196	01/19/2011 02:05:00 AM	(37.798499999999997, -122.27571)
75227	01/19/2011 02:05:00 AM	(37.798596000000003, -122.27569)

... (2742091 rows omitted)

SCRATCH

Phew, that's a lot of data: we can see about 2.7 million license plate reads here.

Let's start by seeing what can be learned about someone, using this data -- assuming you know their license plate.

Stalking Jean Quan

As a warmup, we'll take a look at ex-Mayor Jean Quan's car, and where it has been seen. Her license plate number is 6FCH845. (How did I learn that? Turns out she was in the news for getting $1000 of parking tickets, and the news article (http://www.sfgate.com/bayarea/matier-ross/article/Jean-Quan-Oakland-s-new-mayor-gets-car-booted-3164530.php) included a picture of her car, with the license plate visible. You'd be amazed by what's out there on the Internet...)

```
lprs.where('Plate', '6FCH845')
```

Plate	Timestamp	Location
6FCH845	11/01/2012 09:04:00 AM	(37.79871, -122.276221)
6FCH845	10/24/2012 11:15:00 AM	(37.799695, -122.274868)
6FCH845	10/24/2012 11:01:00 AM	(37.799693, -122.274806)
6FCH845	10/24/2012 10:20:00 AM	(37.799735, -122.274893)
6FCH845	05/08/2014 07:30:00 PM	(37.797558, -122.26935)
6FCH845	12/31/2013 10:09:00 AM	(37.807556, -122.278485)

OK, so her car shows up 6 times in this data set. However, it's hard to make sense of those coordinates. I don't know about you, but I can't read GPS so well.

So, let's work out a way to show where her car has been seen on a map. We'll need to extract the latitude and longitude, as the data isn't quite in the format that the mapping software expects: the mapping software expects the latitude to be in one column and the longitude in another. Let's write some Python code to do that, by splitting the Location string into two pieces: the stuff before the comma (the latitude) and the stuff after (the longitude).

```
def getlatitude(s):
    before, after = s.split(',') # Break it into two parts
    latstring = before[1:] # Get rid of the annoying '('
    return float(latstring) # Convert the string to a number
def getlongitude(s):
    before, after = s.split(',') # Break it into two parts
    longstring = after[1:-1] # Get rid of the ' ' and the ')'
    return float(longstring) # Convert the string to a number
```

Let's test it to make sure it works correctly.

```
getlatitude('(37.797558, -122.26935)')
```

```
37.797558
```

```
getlongitude('(37.797558, -122.26935)')
```

```
-122.26935
```

Good, now we're ready to add these as extra columns to the table.

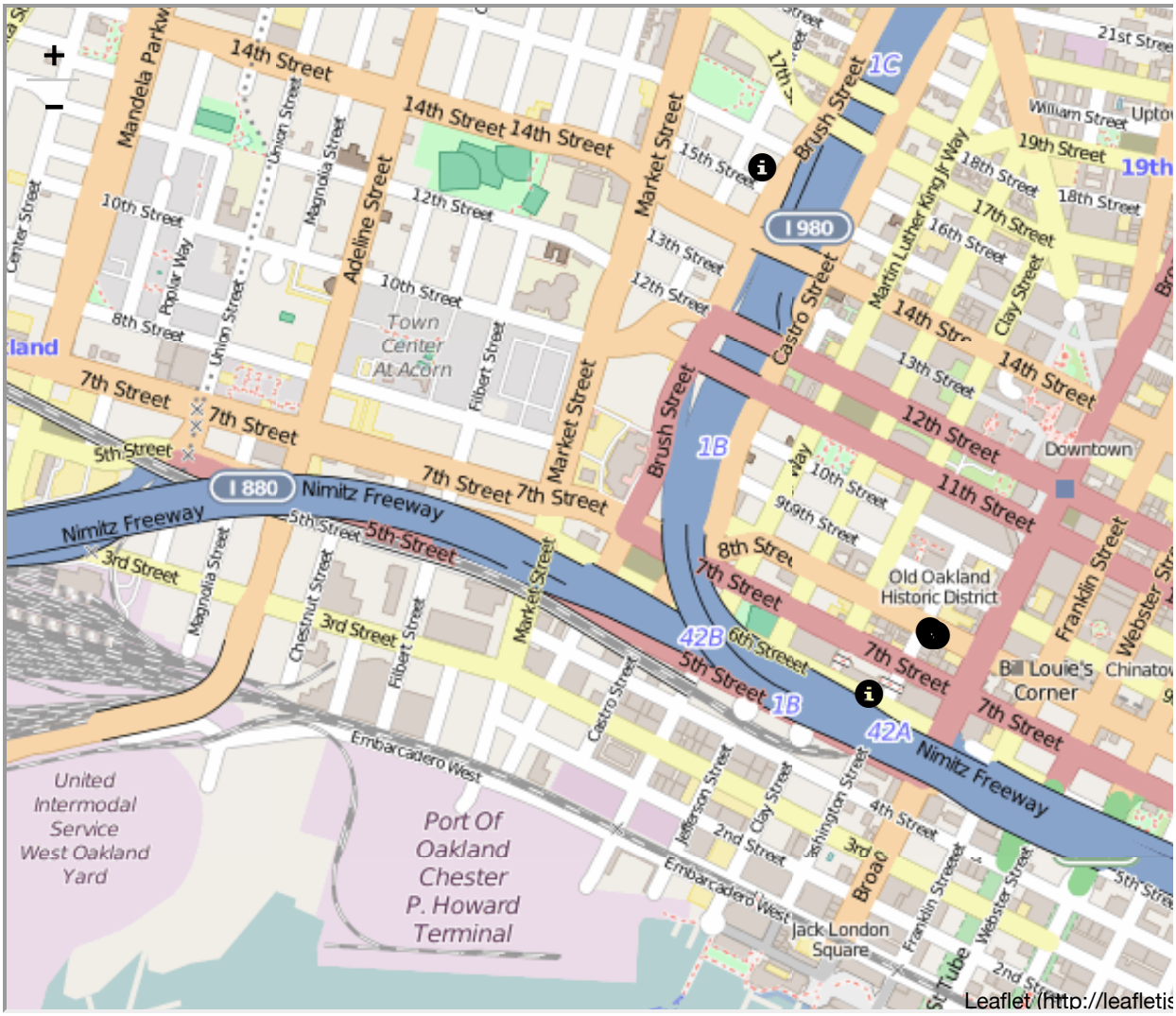
```
lprs['Latitude'] = lprs.apply(getlatitude, 'Location')
lprs['Longitude'] = lprs.apply(getlongitude, 'Location')
lprs = lprs.drop('Location')
lprs
```

Plate	Timestamp	Latitude	Longitude
1275226	01/19/2011 02:06:00 AM	37.7983	-122.276
27529C	01/19/2011 02:06:00 AM	37.7983	-122.276
1158423	01/19/2011 02:06:00 AM	37.7983	-122.276
1273718	01/19/2011 02:06:00 AM	37.7983	-122.276
1077682	01/19/2011 02:06:00 AM	37.7983	-122.276
1214195	01/19/2011 02:06:00 AM	37.7983	-122.276
1062420	01/19/2011 02:06:00 AM	37.7983	-122.276
1319726	01/19/2011 02:05:00 AM	37.7985	-122.276
1214196	01/19/2011 02:05:00 AM	37.7985	-122.276
75227	01/19/2011 02:05:00 AM	37.7986	-122.276

... (2742091 rows omitted)

And at last, we can draw a map with a marker everywhere that her car has been seen.

```
jeanquan = lprs.where('Plate', '6FCH845')
Marker.map(jeanquan['Latitude'], jeanquan['Longitude'], labels=jeanquan['Timestam
```



OK, so it's been seen near the Oakland police department. This should make you suspect we might be getting a bit of a biased sample. Why might the Oakland PD be the most common place where her car is seen? Can you come up with a plausible explanation for this?

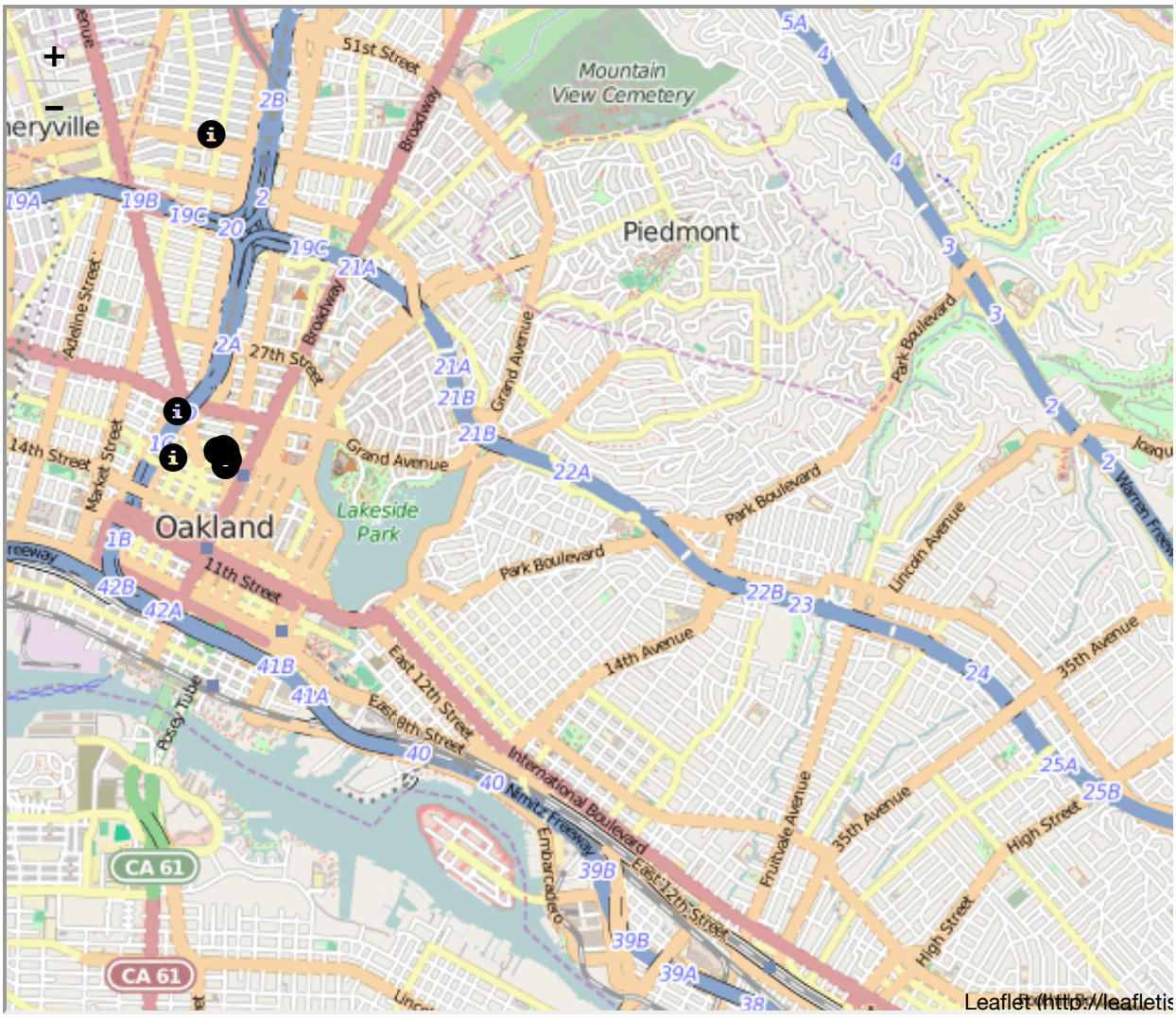
Poking around

Let's try another. And let's see if we can make the map a little more fancy. It'd be nice to distinguish between license plate reads that are seen during the daytime (on a weekday), vs the evening (on a weekday), vs on a weekend. So we'll color-code the markers. To do this, we'll write some Python code to analyze the Timestamp and choose an appropriate color.

```
import datetime
def getcolor(ts):
    t = datetime.datetime.strptime(ts, '%m/%d/%Y %I:%M:%S %p')
    if t.weekday() >= 6:
        return 'green' # Weekend
    if t.hour >= 6 and t.hour <= 17:
        return 'blue' # Weekday daytime
    return 'red' # Weekday evening
lprs['Color'] = lprs.apply(getcolor, 'Timestamp')
```

Now we can check out another license plate, this time with our spiffy color-coding. This one happens to be the car that the city issues to the Fire Chief.

```
t = lprs.where('Plate', '1328354')
Marker.map(t['Latitude'], t['Longitude'], labels=t['Timestamp'], colors=t['Color'
```

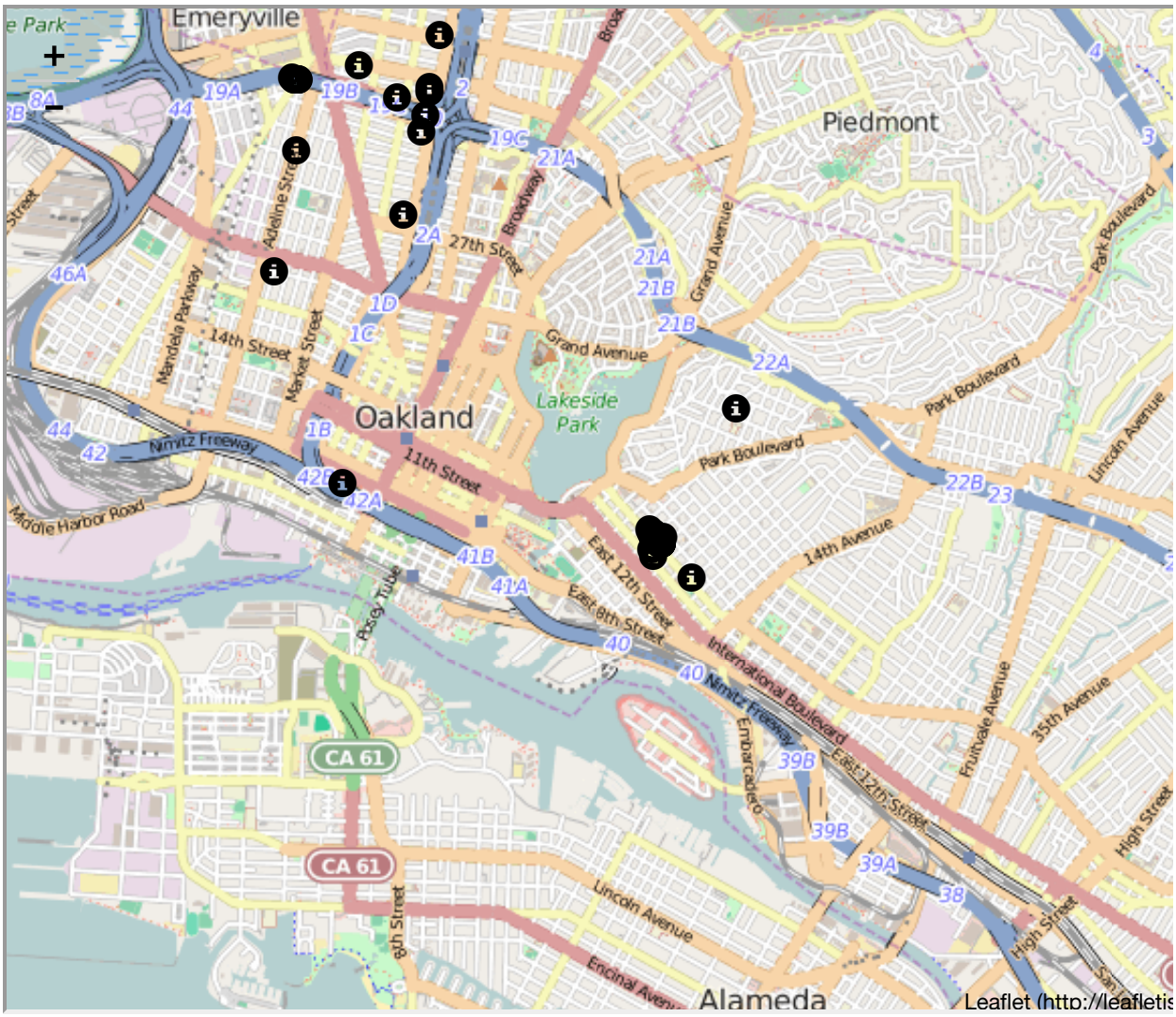


Hmm. We can see a blue cluster in downtown Oakland, where the Fire Chief's car was seen on weekdays during business hours. I bet we've found her office. In fact, if you happen to know downtown Oakland, those are mostly clustered right near City Hall. Also, her car was seen twice in northern Oakland on

weekday evenings. One can only speculate what that indicates. Maybe dinner with a friend? Or running errands? Off to the scene of a fire? Who knows. And then the car has been seen once more, late at night on a weekend, in a residential area in the hills. Her home address, maybe?

Let's look at another.

```
t = lprs.where('Plate', '5AJG153')
Marker.map(t['Latitude'], t['Longitude'], labels=t['Timestamp'], colors=t['Color'
```



What can we tell from this? Looks to me like this person lives on International Blvd and 9th, roughly. On weekdays they've seen in a variety of locations in west Oakland. It's fun to imagine what this might indicate -- delivery person? taxi driver? someone running errands all over the place in west Oakland?

We can look at another:

```
t = lprs.where('Plate', '6UZA652')
Marker.map(t['Latitude'], t['Longitude'], labels=t['Timestamp'], colors=t['Color'
```

What can we learn from this map? First, it's pretty easy to guess where this person lives: 16th and International, or pretty near there. And then we can see them spending some nights and a weekend near Laney College. Did they have an apartment there briefly? A relationship with someone who lived there?

Is anyone else getting a little bit creeped out about this? I think I've had enough of looking at individual people's data.

Inference

As we can see, this kind of data can potentially reveal a fair bit about people. Someone with access to the data can draw inferences. Take a moment to think about what someone might be able to infer from this kind of data.

As we've seen here, it's not too hard to make a pretty good guess at roughly where some lives, from this kind of information: their car is probably parked near their home most nights. Also, it will often be possible to guess where someone works: if they commute into work by car, then on weekdays during business hours, their car is probably parked near their office, so we'll see a clear cluster that indicates where they work.

But it doesn't stop there. If we have enough data, it might also be possible to get a sense of what they like to do during their downtime (do they spend time at the park?). And in some cases the data might reveal that someone is in a relationship and spending nights at someone else's house. That's arguably pretty sensitive stuff.

This gets at one of the challenges with privacy. Data that's collected for one purpose (fighting crime, or something like that) can potentially reveal a lot more. It can allow the owner of the data to draw inferences -- sometimes about things that people would prefer to keep private. And that means that, in a world of "big data", if we're not careful, privacy can be collateral damage.

Mitigation

If we want to protect people's privacy, what can be done about this? That's a lengthy subject. But at risk of over-simplifying, there are a few simple strategies that data owners can take:

1. Minimize the data they have. Collect only what they need, and delete it after it's not needed.
2. Control who has access to the sensitive data. Perhaps only a handful of trusted insiders need access; if so, then one can lock down the data so only they have access to it. One can also log all access, to deter misuse.
3. Anonymize the data, so it can't be linked back to the individual who it is about. Unfortunately, this is often harder than it sounds.
4. Engage with stakeholders. Provide transparency, to try to avoid people being taken by surprise. Give individuals a way to see what data has been collected about them. Give people a way to opt out and have their data be deleted, if they wish. Engage in a discussion about values, and tell people what steps you are taking to protect them from unwanted consequences.

This only scratches the surface of the subject. My main goal in this lecture was to make you aware of privacy concerns, so that if you are ever a steward of a large data set, you can think about how to protect people's data and use it responsibly.