Spatial data science with R

Introducing the osmenrich package

ODISSEI Social Data Science team

A 3-part journey

Part 1.
ODISSEI and the SoDa team.
Helen Lam

Part 2.
Spatial data in R with simple features & the osmenrich package.
Leonardo Vida

Part 3.
Spatial regression with features from osmenrich.
Erik-Jan van Kesteren

Part 1.

ODISSEI and the SoDa team

ODISSEI

- Open Data and Research Infrastructure for Social Science and Economics
- Consortium of 34 Dutch scientific organizations
- In Utrecht:
  - Faculty of Geosciences
  - Faculty of Social Sciences
  - Faculty of Law, Economics and Governance (REBO)
- Support researchers by e.g., access to CBS microdata or LISS panel, use of the supercomputer, support in computational challenges

The SoDa team

- ODISSEI Social Data Science team (SoDa)
- Providing data science support to social scientists at ODISSEI institutes

SoDa team members

- Daniel Oberski
- Erik-Jan van Kesteren
- Leonardo Vida
- Helen Lam
- Jonathan de Bruin

Activities of the team

Data acquisition
New sources

Data analysis
New methods

Interpretation
New insights

Communication
New impact

Where to find us

On the website of ODISSEI
https://odissei-data.nl/soda

On GitHub
https://github.com/sodascience
https://github.com/sodascience/osmenrich
https://github.com/sodascience/presentation-osmenrich-sig
Your contributions, questions, and issue reports are greatly appreciated!

Via email
soda@odissei-data.nl
e.vankesteren1@uu.nl (also via Teams!)

Part 2.

Spatial data in R with simple features & the `osmenrich` package

Simple Features

Illustration (c) by Allison Horst

Simple Features

- A **simple feature** is a standard representation of physical objects that has spatial and non-spatial attributes
- **17** simple feature types (like *point*, *line*, *polygon*)
- Supported by many databases and software

Simple Features in R (sf)

- In R simple features are implemented in the package `sf`
- Provides simple features in `data.frames` or `tibbles` with a geometry list-column

```
library(sf)
# > Linking to GEOS 3.7.2, GDAL 2.4.2, PROJ 5.2.0
```

Simple Features and `data.frames`

- A common use case is working with datasets consisting **of sets of simple features** with attributes

```
nc_data_path <- system.file("shape/nc.shp", package = "sf")

selected_data <- st_read(nc_data_path) %>%
        select(CNTY_ID, NAME, BIR74)
```

Simple Features and `data.frames`

```
selected_data

## Simple feature collection with 100 features and 3 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -84.32 ymin: 33.88 xmax: -75.45 ymax:36.58
## geographic CRS: NAD27
## First 3 features:
##    CNTY_ID        NAME BIR74                   geometry
## 1     1825        Ashe  1091 MULTIPOLYGON (((-81.47276 3...
## 2     1827   Alleghany   487 MULTIPOLYGON (((-81.23989 3...
## 3     1828       Surry  3188 MULTIPOLYGON (((-80.45634 3...
## 4     1831   Currituck   508 MULTIPOLYGON (((-76.00897 3...
## 5     1832 Northampton  1421 MULTIPOLYGON (((-77.21767 3...
## 6     1833    Hertford  1452 MULTIPOLYGON (((-76.74506 3...
```

Simple Features and data.frames

```
selected_data

## Simple feature collection with 100 features and 3 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -84.32 ymin: 33.88 xmax: -75.45 ymax:36.58
## geographic CRS: NAD27
## First 6 features:
##    CNTY_ID        NAME BIR74                       geometry
## 1     1825        Ashe  1091 MULTIPOLYGON (((-81.47276 3...
## 2     1827   Alleghany   487 MULTIPOLYGON (((-81.23989 3...
## 3     1828       Surry  3188 MULTIPOLYGON (((-80.45634 3...
## 4     1831   Currituck   508 MULTIPOLYGON (((-76.00897 3...
## 5     1832 Northampton  1421 MULTIPOLYGON (((-77.21767 3...
## 6     1833    Hertford  1452 MULTIPOLYGON (((-76.74506 3...
```

Simple feature

Simple feature geometry list-column

Simple feature geometry

OpenStreetMap (OSM)

- Like Wikipedia but for Maps (*openstreetmap.org*)
- OSM is often needed to work with spatial or geographic data as it provides **free and open map data**

OpenStreetMap and `sf`

- **To extract** data from OSM using R you can use `osmdata`

- `osmdata` allows you to retrieve OSM features in `sf` format

```
library("sf")
library("osmdata")
library("dplyr")
```

Example extraction of Trees

- Extract the boundaries of Utrecht (“*bbox*”) using `osmdata`

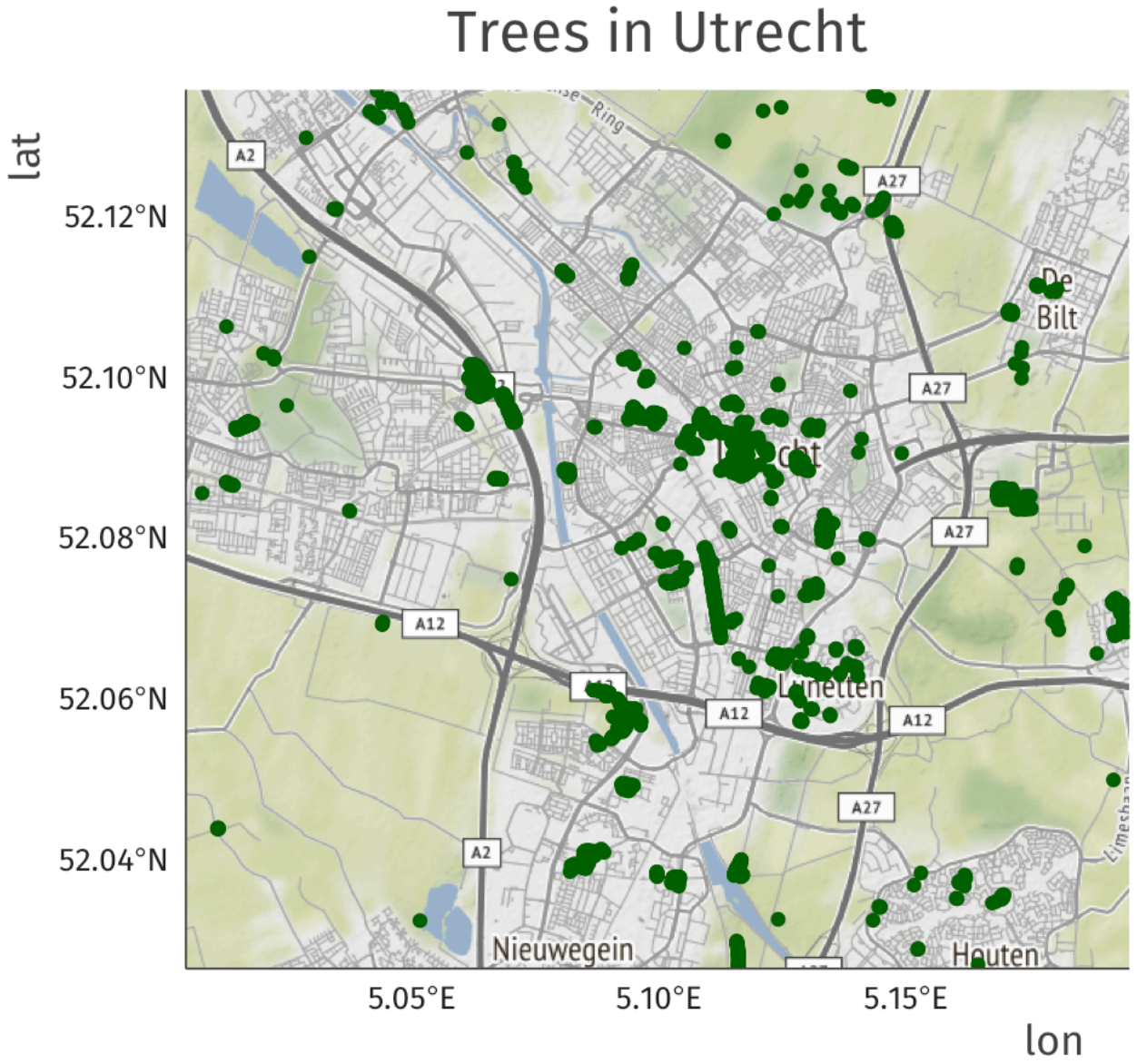
```
utrecht_query <- osmdata::opq(bbox = “utrecht nl”)

## “52.026282,5.0041822,52.1356715,5.195155”
```

- With the boundary of Utrecht it is possible to extract trees from OSM

```
trees_sf <- utrecht_query %>%
  osmdata::add_osm_feature(key = "natural", value = "tree") %>%
  osmdata::osmdata_sf()
```

Plot Trees

Use `ggplot` to plot the points downloaded from OSM on a map of Utrecht



How to integrate this data with existing data?

This is quite a lot of work!

osmenrich

What is `osmenrich`?

- A package to **enrich** geocoded data (*lat*/*lon*) with OpenStreetMap data
- Makes data enrichment as **intuitive and easy** as possible
- Designed to work with the `sf` package

How to install `osmenrich`

- To install from GitHub you need the “`remotes`” package:

```
install.packages("remotes")
```

- To install and load **`osmenrich`**:

```
remotes::install_github("sodascience/osmenrich")
```

```
library(osmenrich)
```

Example: sparrows dataset

- *Two sparrows want to know **how many trees** there are in a **1km radius** from them*
- Let's create a dataset with Sparrow Alice and Bob and transform it into an `sf` dataset

```
library(tidyverse)
library(sf)

sf_sparrows <-
    tribble(
        ~sparrow,         ~lat,  ~lon,
        "Sparrow_Alice", 52.12, 5.09,
        "Sparrow_Bob",   52.13, 5.08,
    ) %>%
    sf::st_as_sf(
        coords = c("lon", "lat"),
        crs = 4326
    )
```

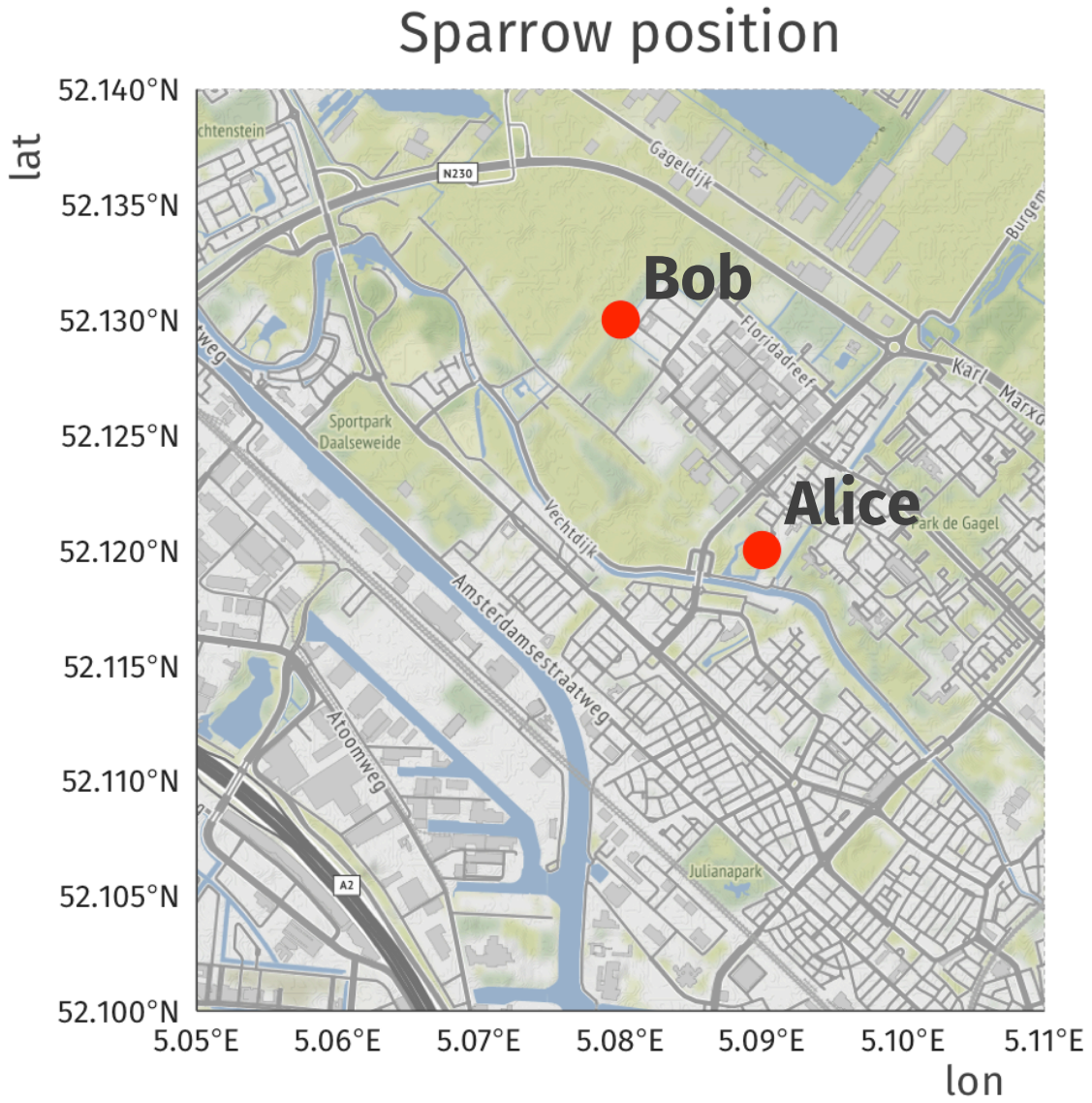
Example: sparrows dataset

- Printing the example dataset shows the typical `sf` view

```
sf_sparrows
```

```
## Simple feature collection with 2 features and 3 fields
## geometry type: POINT
## dimension: XY
## bbox: xmin: 5.08 ymin: 52.12 xmax: 5.09 ymax: 52.13
## CRS: EPSG:4326
## # A tibble: 2 x 2
##   sparrow        geometry
## * <chr>          <POINT [°]>
## 1 Sparrow_Alice  (5.09 52.12)
## 2 Sparrow_Bob    (5.08 52.13)
```

Positions of the sparrows



Enriching sparrows dataset with trees

- **`osmenrich`** is used to count the trees around the sparrows and add the count to the dataset

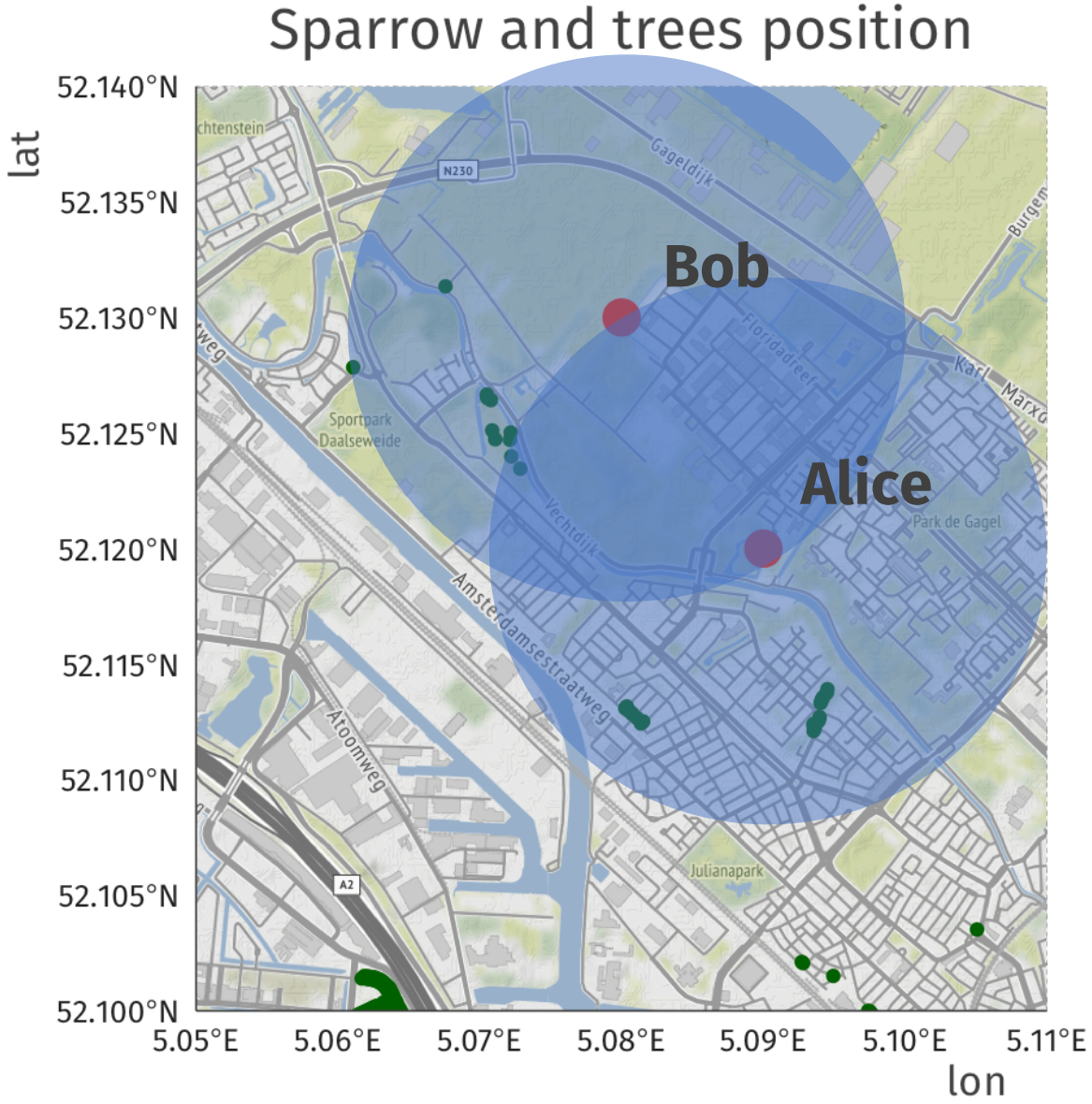
```
sf_sparrows_enriched <- sf_sparrows %>%
  osmenrich::enrich_osm(
    name = "n_trees",
    key = "natural",
    value = "trees",
    r = 1000
  )
#> Downloading data for n_trees... Done.
#> Downloaded 35 points, 0 lines, 0 polygons, 0 mlines, 0 mpolygons.
#> Computing distance matrix for n_trees...Done.
#> Adding n_trees to data.
```

Enriching sparrows dataset with trees

- Print the enriched example dataset to look at the results

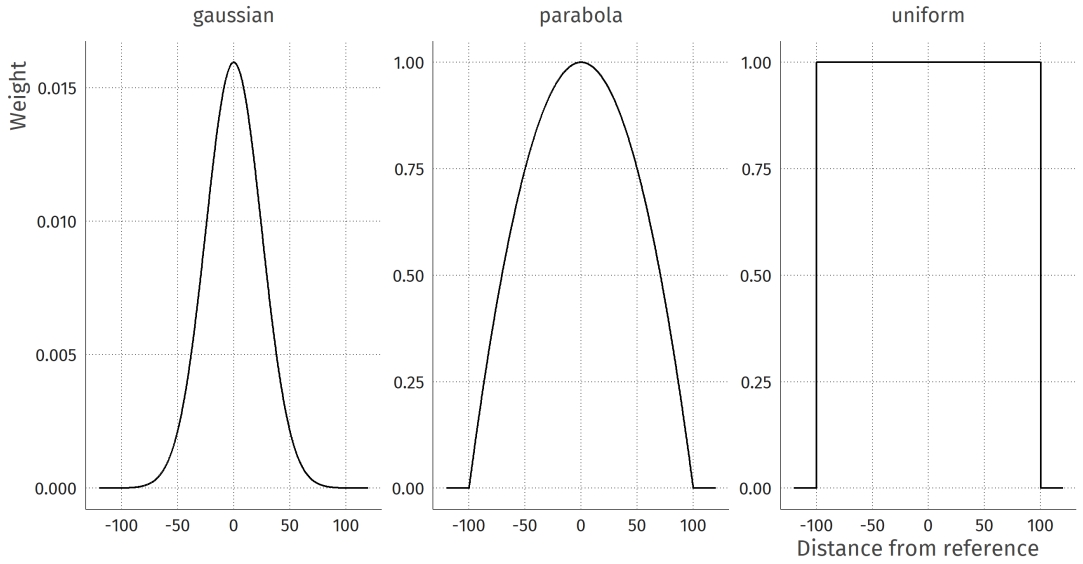
```
sf_sparrows_enriched

#> Simple feature collection with 2 features and 4 fields
#> geometry type: POINT
#> dimension: XY
#> bbox: xmin: 5.08 ymin: 52.12 xmax: 5.09 ymax: 52.13
#> CRS: EPSG:4326
#> A tibble: 2 x 3
#>   sparrow          geometry     n_trees
#> * <chr>            <POINT [°]>  <int>
#> 1 Sparrow_Alice    (5.09 52.12) 14
#> 2 Sparrow_Bob      (5.08 52.13) 10
```

Plot the enriched dataset



Special features of `osmenrich`

- **`enrich_osm()`** has a kernel argument that enables users to define distance-based weights in aggregation of the retrieved data



- **`osmenrich`** can also retrieve driving/cycling/walking distances and durations from points in the dataset (e.g. *sparrows*) to features (e.g. *trees*)

`osmenrich` is still in development!

- Actively searching for new projects that use **`osmenrich`**
- Any contribution and question on our GitHub page is welcome
- https://github.com/sodascience/osmenrich

Part 3.

Spatial regression with features from osmenrich

Disclaimer: this is an example analysis,
NOT a careful research project!

Running example: gierzwaluw

Open data from Utrecht municipality
https://ckan.dataplatform.nl/dataset/8ceaae10-fb90-4ec0-961c-ef02691bb861

Gierzwaluwinventarisatie 2014
Counting the nests of the common swift
CC-0 license

Variables
Nest count (1-4)
Latitude
Longitude
Address information
Detailed description of the nest

Full code available at:

https://github.com/sodascience/presentation-osmenrich-sig

Loading the data

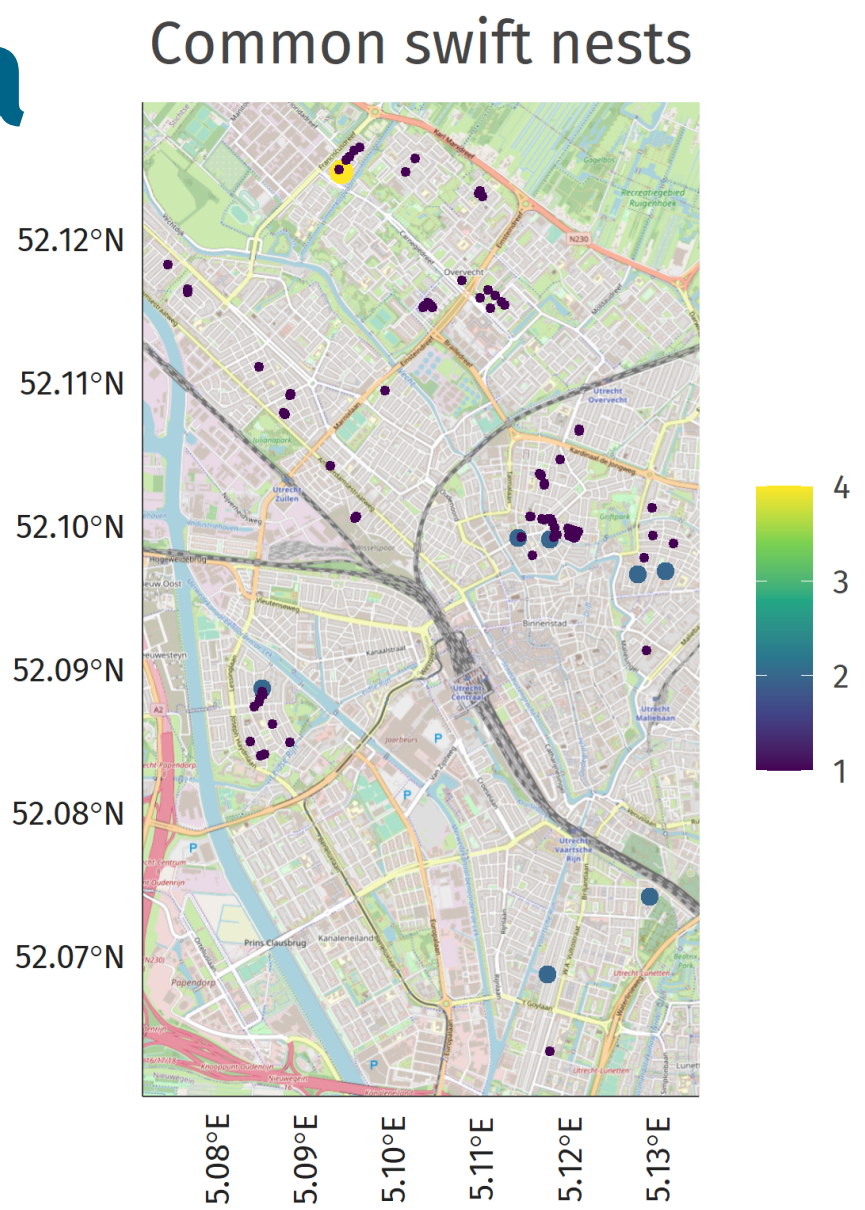
```
library(tidyverse)
library(sf)
data_url <- "https://ckan.dataplatform.nl/dataset/8ceaae10-fb90-4ec0-961c-ef02691bb861/resource/baae4cde-cf33-416b-aa4e-d0fba160eed9/download/gierzwaluwinventarisatie2014.csv"

bird_sf <-
  read_csv(data_url) %>%
  drop_na(latitude, longitude, `aantal nesten`) %>%
  st_as_sf(coords = c("longitude", "latitude"), crs = 4326) %>%
  select(nestcount = "aantal nesten", geometry)
```

Loading the data

```
Simple feature collection with 99 features and 1 field
geometry type:  POINT
dimension:      XY
bbox:           xmin: 5.075877 ymin: 52.06334 xmax: 5.133265 ymax: 52.12635
geographic CRS: WGS 84
# A tibble: 99 x 2
   nestcount            geometry
       <dbl>         <POINT [°]>
 1         1  (5.086195 52.0884)
 2         2 (5.086602 52.08866)
 3         1  (5.086619 52.0884)
 4         1 (5.086604 52.08837)
# ... with 95 more rows
```

The gierzwaluw data

```
library(ggspatial)

plot_bird <-
  ggplot(bird_sf) +
  annotation_map_tile() +
  geom_sf() +
  labs(title = "Common swift nests")
```



Sneaky trick

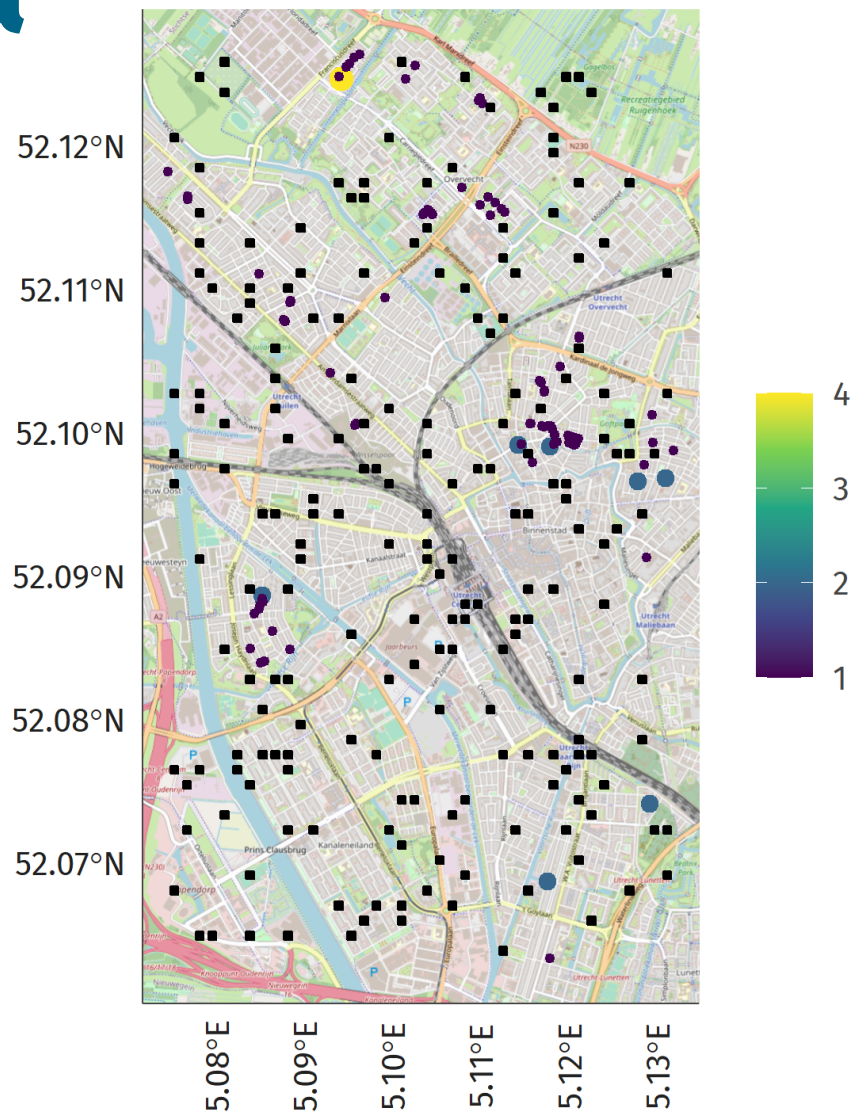
- We want to predict nestcount
- But dataset only contains observed nests
- Simplifying assumption: data contains all nests in Utrecht

- i.e. all other locations have 0 nests!
- Randomly sample points with 0 nestcount
- = Data augmentation

- You should use a more sophisticated model to get around this

The gierzwaluw data

```
bird_sf <- bind_rows(bird_sf, nonbird_sf)

plot_bird <-
  ggplot(bird_sf) +
  annotation_map_tile() +
  geom_sf() +
  labs(title = "Common swift nests")
```



I’m an ecologist now
My expectations

Nesting behaviour is spatially correlated
We are more likely to find a nest close to other nests

Common swifts benefit from commercial activity, but not too much
The birds need food but also they need peace. They are peaceful birds.

Nature is important to the common swift
The birds need trees to do bird things like sitting and nesting

PS. I am not an ecologist
But let’s test these expectations anyway!

We need more data to test our expectations about nesting behaviour

We can gather a lot of this data from open sources such as OSM

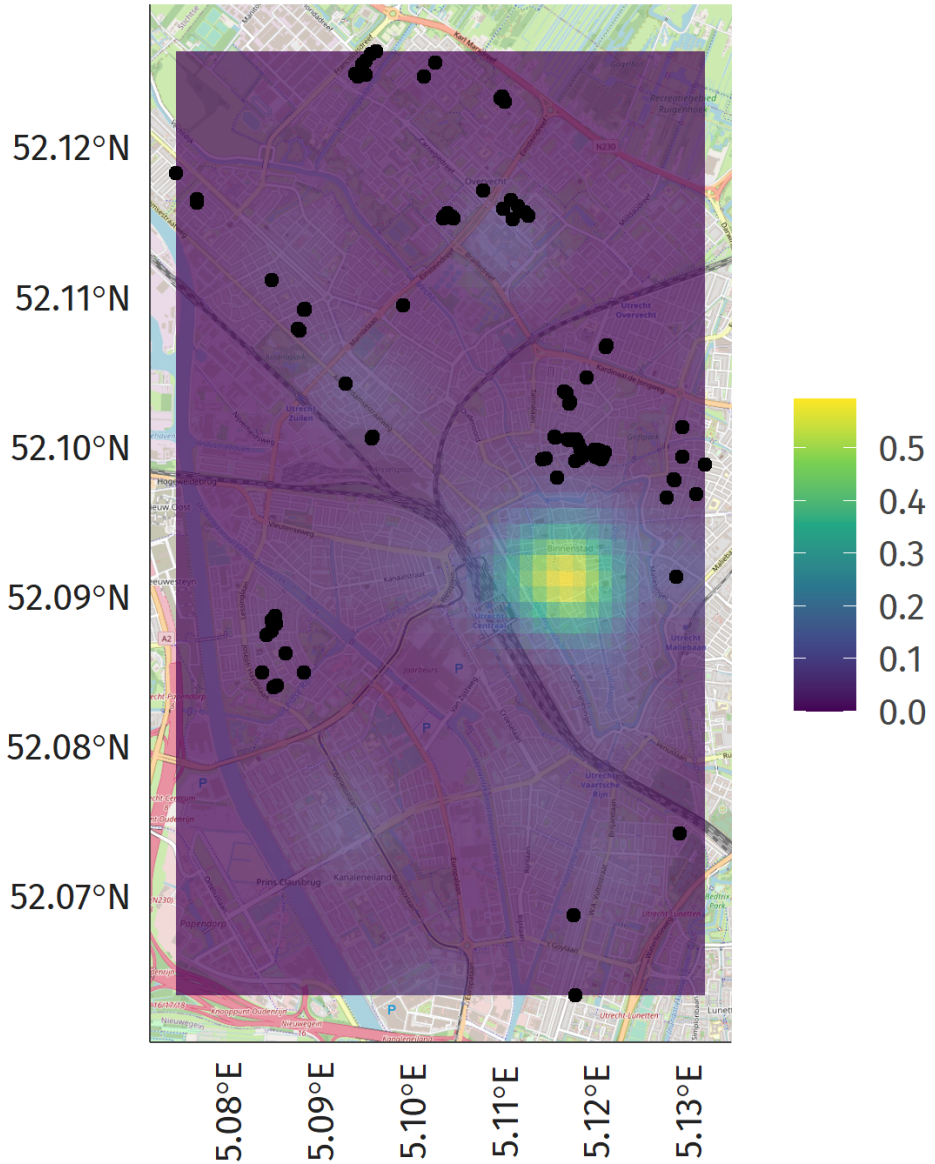
Data enrichment

```
library(osmenrich)

bird_sf <-
  bird_sf %>%
  enrich_osm(
    name   = "commercial_activity_1km",
    key    = "shop",
    kernel = "gaussian",
    r      = 1000
  )
```

Data enrichment

```
Simple feature collection with 299 features and 2 fields
geometry type:  POINT
dimension:      XY
bbox:           xmin: 5.075877 ymin: 52.06334 xmax: 5.133265 ymax: 52.12635
geographic CRS: WGS 84
# A tibble: 299 x 3
   nestcount commercial_activity_1km            geometry
       <dbl>                   <dbl>         <POINT [°]>
 1         1                  0.0167   (5.086195 52.0884)
 2         2                  0.0165  (5.086602 52.08866)
 3         1                  0.0167   (5.086619 52.0884)
 4         1                  0.0167  (5.086604 52.08837)
# ... with 295 more rows
```

Commercial activity



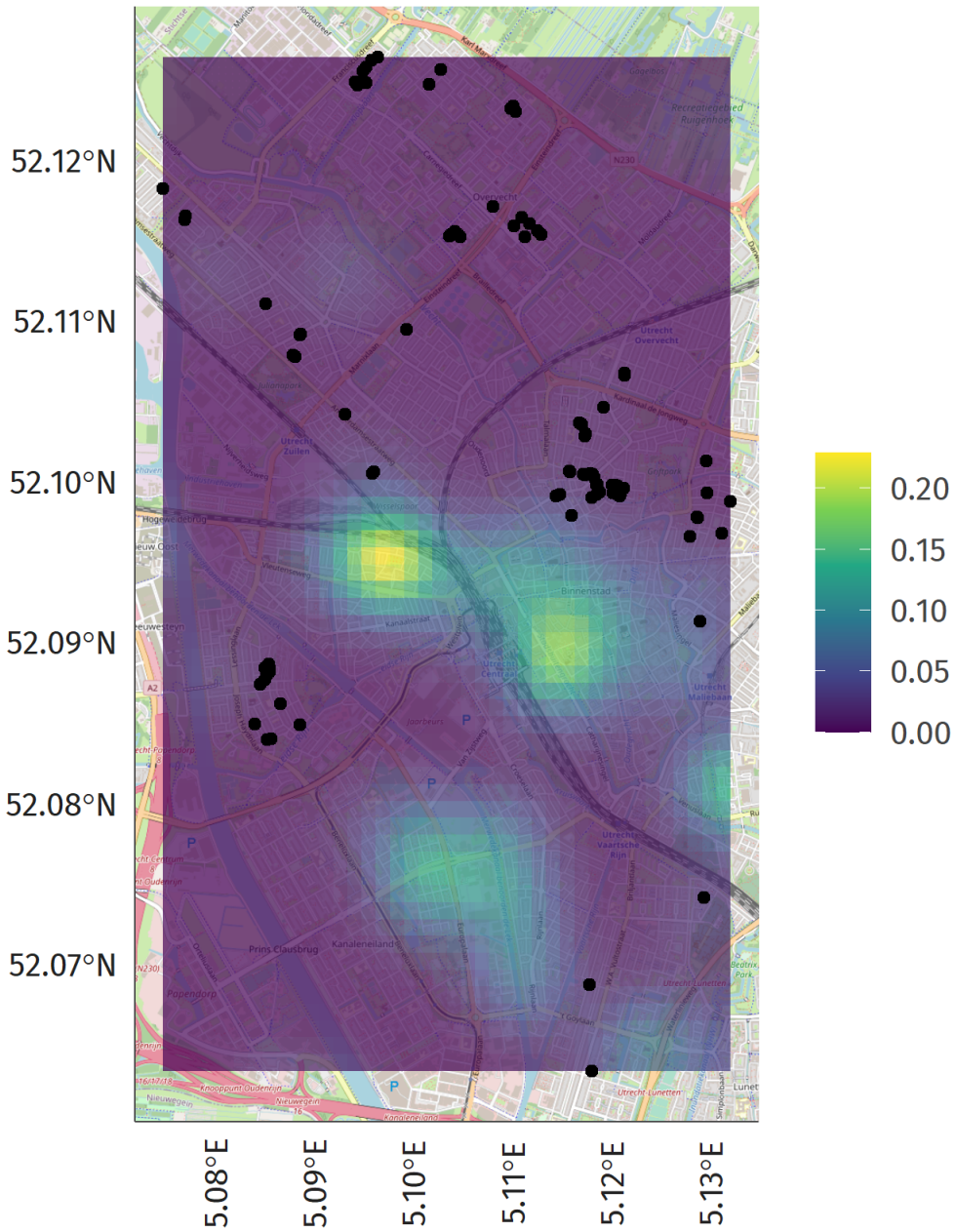
Data enrichment

```
bird_sf <-
  bird_sf %>%
  enrich_osm(
    name   = "tree_1km",
    key    = "natural",
    value  = "tree",
    kernel = "gaussian",
    r      = 1000
  )
```

Data enrichment

```
Simple feature collection with 299 features and 3 fields
geometry type:  POINT
dimension:      XY
bbox:           xmin: 5.075877 ymin: 52.06334 xmax: 5.133265 ymax: 52.12635
geographic CRS: WGS 84
# A tibble: 299 x 4
   nestcount commercial_activity_1km  tree_1km            geometry
       <dbl>                   <dbl>     <dbl>         <POINT [°]>
 1         1                  0.0167   0.00799   (5.086195 52.0884)
 2         2                  0.0165   0.00692  (5.086602 52.08866)
 3         1                  0.0167   0.00689   (5.086619 52.0884)
 4         1                  0.0167   0.00693  (5.086604 52.08837)
# ... with 295 more rows
```

Trees


52.12°N
52.11°N
52.10°N
52.09°N
52.08°N
52.07°N
5.08°E
5.09°E
5.10°E
5.11°E
5.12°E
5.13°E
0.20
0.15
0.10
0.05
0.00

Now we have our data

Regression model

Count outcome

- Zero-inflated poisson regression with identity link function

Predictors: generalized additive model

- Bivariate spline for the spatial component (smooth spatial variation)
- Quadratic effect of commercial activity
- Linear effect of trees

NB: There are many options here: gaussian process regression, regression kriging, other smoothing techniques, ...

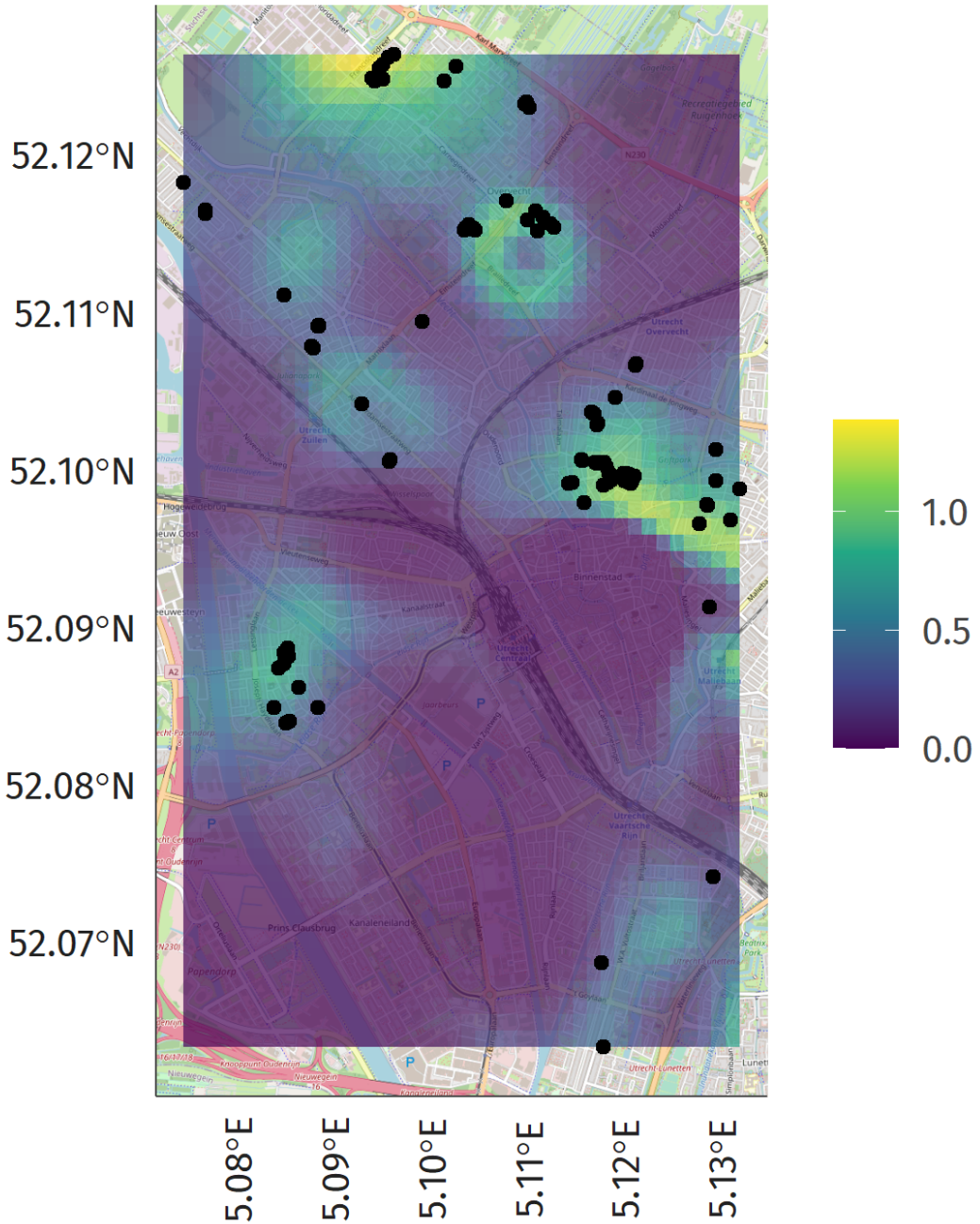
Regression model

```
library(mgcv)

fit <- gam(
  nestcount ~
    te(x, y, bs = "ts") +
    poly(commercial_activity_1km, 2) +
    tree_1km,
  family = "ziP",
  data = sf_to_df(bird_sf)
)
```

Regression model

```
Parametric coefficients:
                                    Estimate Std. Error z value Pr(>|z|)
(Intercept)                          -5.4450     0.9183  -5.929 3.04e-09 ***
poly(commercial_activity_1km, 2)1 -241.9024    85.0221  -2.845  0.00444 **
poly(commercial_activity_1km, 2)2 -154.0026    47.7631  -3.224  0.00126 **
tree_300m                          -27.3748    14.8989  -1.837  0.06615 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
          edf Ref.df Chi.sq  p-value
te(x,y) 14.97     24  86.81 8.59e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Nest count


52.12°N
52.11°N
52.10°N
52.09°N
52.08°N
52.07°N
5.08°E
5.09°E
5.10°E
5.11°E
5.12°E
5.13°E
1.0
0.5
0.0

Conclusion

- Spatial observational data can be enriched easily using sf and osmenrich
- Spatial models can then be used to test expectations about behaviour in the environment
- Spatial predictions from these models can guide future data collection efforts

Questions?

Thank you!