Research Project

Deekshitha Goud Podeti

2025-08-15

```
pacman::p_load(tidyverse, dplyr, readr, here, sf, tmap, corrplot)
packageVersion("tmap")
```

```
## [1] '4.1'
```

loading the dataset.

```
# Load packages
library(tidyverse)
library(sf)
library(tmap)
library(here)

# Step 1: Load and joining data
year_msoa_grocery <- read_csv(here("data/year_msoa_grocery.csv")) %>%
  rename(msoa_code = area_id)
```

```
## Rows: 983 Columns: 202
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): area_id
## dbl (201): weight, weight_perc2.5, weight_perc25, weight_perc50, weight_perc...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
msoas <- st_read(here("data/statistical-gis-boundaries-london/statistical-gis-boundaries-london/ESRI/MS
  rename(msoa_code = MSOA11CD, msoa_name = MSOA11NM)
```

```
## Reading layer `MSOA_2011_London_gen_MHW' from data source
##   `\\uofa\users$\users9\a1907979\Research project\Data\statistical-gis-boundaries-london\statistical
##   using driver `ESRI Shapefile'
## Simple feature collection with 983 features and 12 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 503574.2 ymin: 155850.8 xmax: 561956.7 ymax: 200933.6
## Projected CRS: OSGB36 / British National Grid
```

```r
tesco_and_msoas <- left_join(msoas, year_msoa_grocery, by = "msoa_code")
```

Creating a Model

```r
full_model <- lm(
  energy_density ~ avg_age + people_per_sq_km + `age_65+` + man_day + transaction_days +
    num_transactions + population + male + female +
    f_readymade + f_wine + f_soft_drinks + f_fruit_veg + f_sweets +
    f_meat_red + f_dairy + f_eggs + f_fish + f_grains + f_poultry + f_water,
  data = tesco_and_msoas
)
summary(full_model)
```

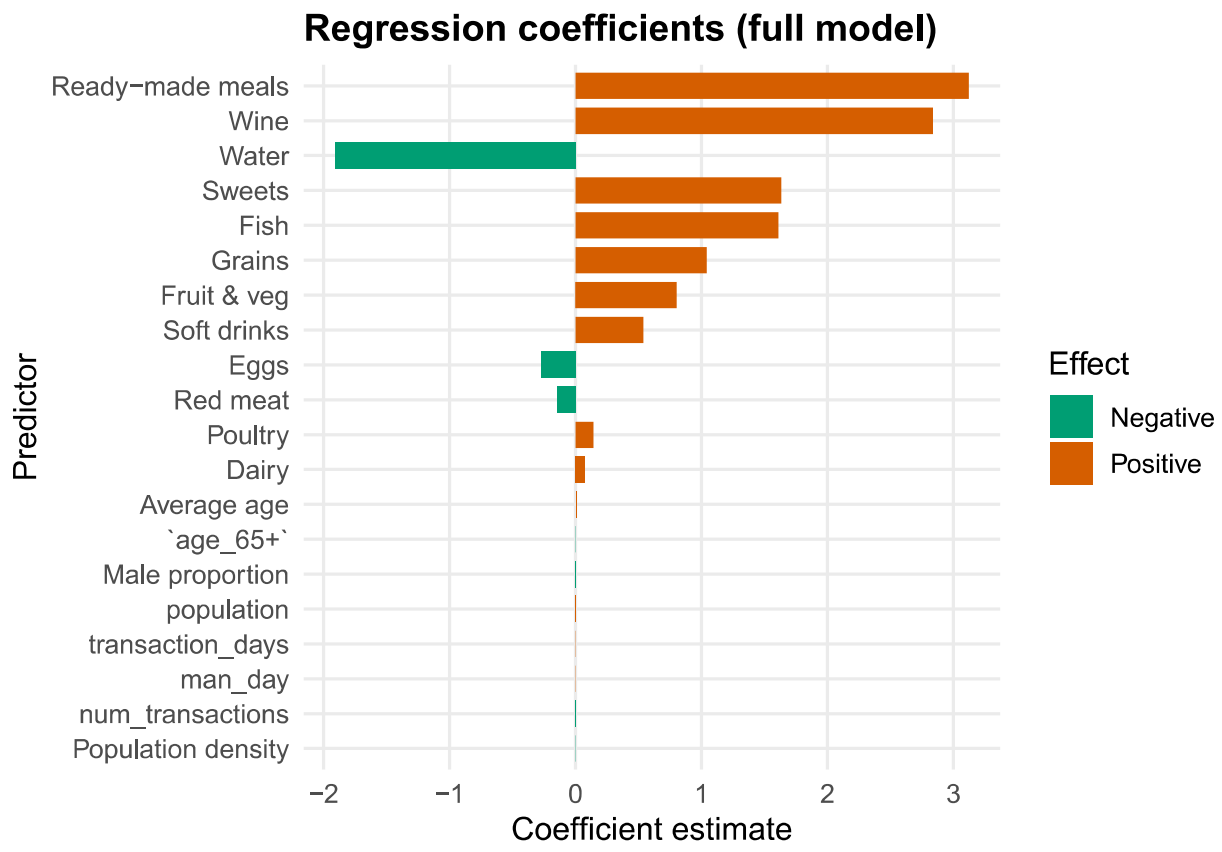
```
##
## Call:
## lm(formula = energy_density ~ avg_age + people_per_sq_km + `age_65+` +
##     man_day + transaction_days + num_transactions + population +
##     male + female + f_readymade + f_wine + f_soft_drinks + f_fruit_veg +
##     f_sweets + f_meat_red + f_dairy + f_eggs + f_fish + f_grains +
##     f_poultry + f_water, data = tesco_and_msoas)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.165142 -0.014288 -0.000295  0.014284  0.139413
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -5.998e-01  9.392e-01  -0.639  0.52324
## avg_age           5.861e-03  9.672e-04   6.059 1.96e-09 ***
## people_per_sq_km -3.128e-09  2.323e-07  -0.013  0.98926
## `age_65+`        -6.354e-05  8.378e-06  -7.584 7.88e-14 ***
## man_day           1.450e-07  7.766e-08   1.867  0.06223 .
## transaction_days  1.239e-05  2.553e-03   0.005  0.99613
## num_transactions -1.637e-08  5.856e-09  -2.795  0.00529 **
## population        2.645e-05  3.777e-06   7.003 4.70e-12 ***
## male             -3.747e-05  6.316e-06  -5.932 4.16e-09 ***
## female                   NA         NA      NA       NA
## f_readymade       3.120e+00  1.609e-01  19.393  < 2e-16 ***
## f_wine            2.835e+00  2.884e-01   9.831  < 2e-16 ***
## f_soft_drinks     5.381e-01  1.945e-01   2.766  0.00578 **
## f_fruit_veg       8.040e-01  1.546e-01   5.202 2.41e-07 ***
## f_sweets          1.632e+00  1.729e-01   9.437  < 2e-16 ***
## f_meat_red       -1.483e-01  2.786e-01  -0.532  0.59465
## f_dairy           7.187e-02  1.872e-01   0.384  0.70110
## f_eggs           -2.723e-01  6.144e-01  -0.443  0.65772
## f_fish            1.608e+00  3.927e-01   4.096 4.56e-05 ***
## f_grains          1.040e+00  1.783e-01   5.833 7.41e-09 ***
## f_poultry         1.411e-01  4.453e-01   0.317  0.75149
## f_water          -1.906e+00  2.562e-01  -7.441 2.22e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.02626 on 962 degrees of freedom
## Multiple R-squared:  0.8793, Adjusted R-squared:  0.8768
## F-statistic: 350.3 on 20 and 962 DF,  p-value: < 2.2e-16
```

Identifying Positive predictors.

```r
# --- Libraries
library(dplyr)
library(broom)
library(ggplot2)

# --- Create output folder (if it doesn't exist)
dir.create("figures", showWarnings = FALSE)

# --- Tidy coefficients
model_coef <- tidy(full_model)

model_coef_clean <- model_coef %>%
  filter(!is.na(estimate), term != "(Intercept)") %>%
  mutate(
    label = dplyr::recode(term,
      f_readymade   = "Ready-made meals",
      f_wine        = "Wine",
      f_sweets      = "Sweets",
      f_fish        = "Fish",
      f_grains      = "Grains",
      f_fruit_veg   = "Fruit & veg",
      f_soft_drinks = "Soft drinks",
      f_poultry     = "Poultry",
      f_dairy       = "Dairy",
      f_eggs        = "Eggs",
      f_meat_red    = "Red meat",
      f_water       = "Water",
      avg_age       = "Average age",
      people_per_sq_km = "Population density",
      male          = "Male proportion",
      .default      = term
    ),
    direction = ifelse(estimate > 0, "Positive", "Negative"),
    label = reorder(label, abs(estimate))
  )

# --- Build the plot
p_coef <- ggplot(model_coef_clean, aes(x = label, y = estimate, fill = direction)) +
  geom_col(width = 0.75) +
  coord_flip() +
  scale_fill_manual(values = c("Positive" = "#D55E00", "Negative" = "#009E73")) +
  labs(
    title = "Regression coefficients (full model)",
    x = "Predictor",
    y = "Coefficient estimate",
    fill = "Effect"
  ) +
```

```
  theme_minimal(base_size = 12) +
  theme(
    plot.title = element_text(face = "bold"),
    legend.position = "right",
    panel.grid.minor = element_blank()
  )

# --- SHOW the plot in the knitted document
p_coef
```




```
# --- SAVE the plot to files for LaTeX/slides
ggsave("figures/coe.pdf", plot = p_coef, width = 7, height = 5, device = cairo_pdf)
ggsave("figures/coe.png", plot = p_coef, width = 7, height = 5, dpi = 300)
```

Creating Spaital maps for positive predectors.

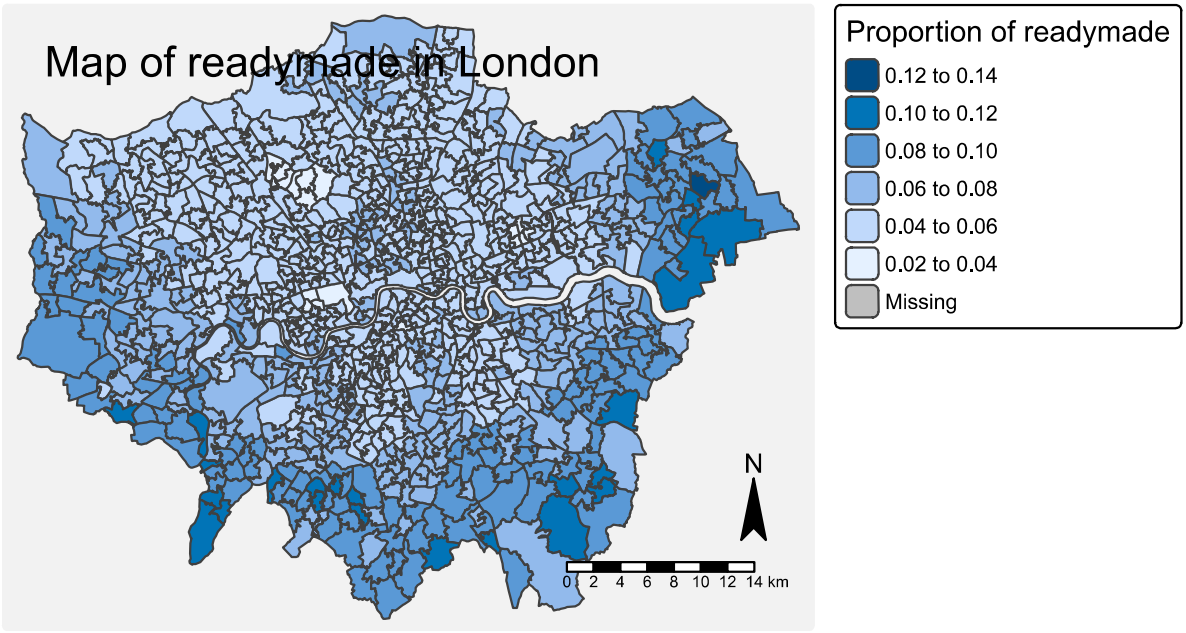
```
library(tidyverse)
library(sf)
library(tmap)

# List of variables with positive coefficients, including avg_age
positive_vars <- c("f_readymade", "f_wine", "f_sweets", "f_fish", "f_grains",
                   "f_fruit_veg", "f_soft_drinks", "f_poultry", "f_dairy", "avg_age")

# Replace NA values with 0 (optional - only if needed to prevent plot errors)
```

```r
tesco_and_msoas <- tesco_and_msoas %>%
  mutate(across(all_of(positive_vars), ~replace_na(., 0)))

# Set tmap mode to plot
tmap_mode("plot")
```

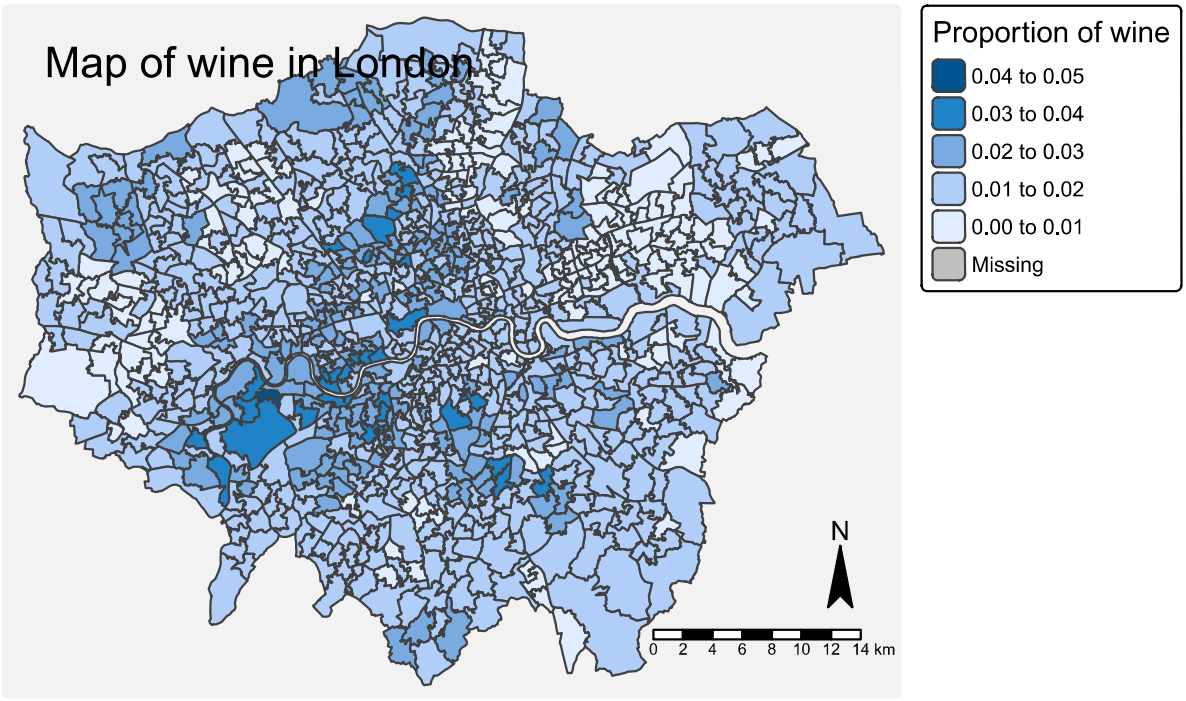
```
## i tmap mode set to "plot".
```

```r
# Loop to generate maps for each variable
for (var in positive_vars) {
  print(
    tm_shape(tesco_and_msoas[!is.na(tesco_and_msoas[[var]]), ]) +
      tm_polygons(
        fill = var,
        fill.palette = "Blues",
        title = paste("Proportion of", gsub("f_", "", var)),
        legend.reverse = TRUE
      ) +
      tm_layout(
        title = paste("Map of", gsub("f_", "", var), "in London"),
        frame = FALSE,
        bg.color = "grey95",
        legend.outside = TRUE
      ) +
      tm_compass() +
      tm_scale_bar()
  )
}
```

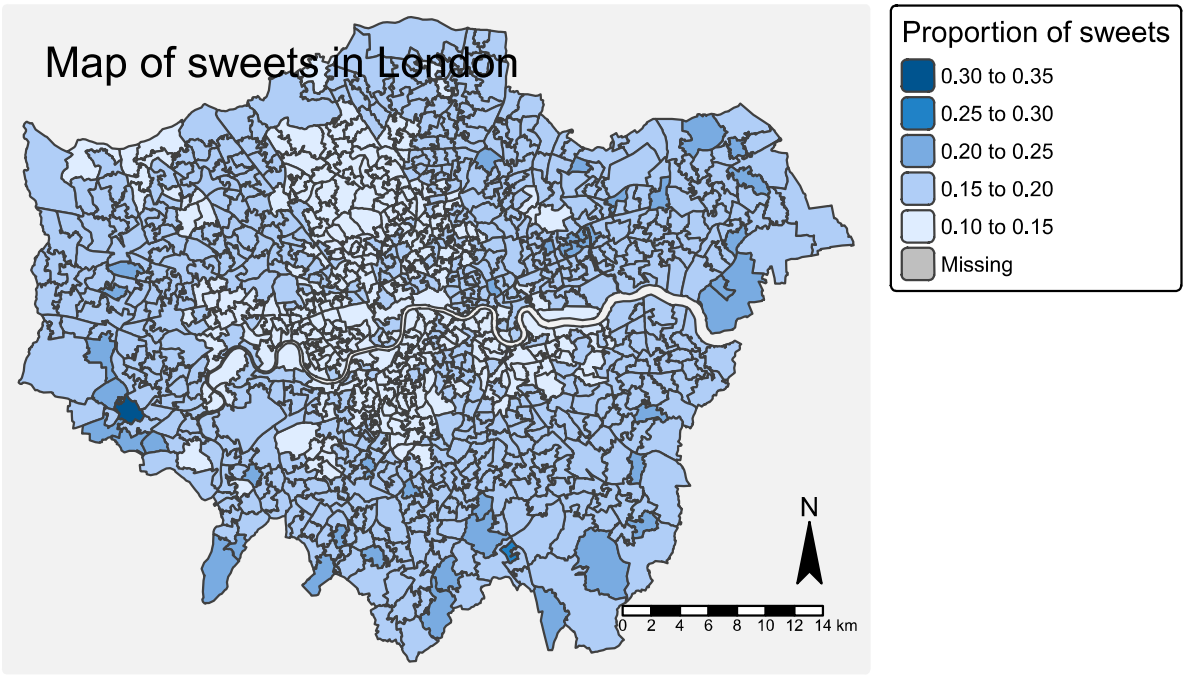
```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```



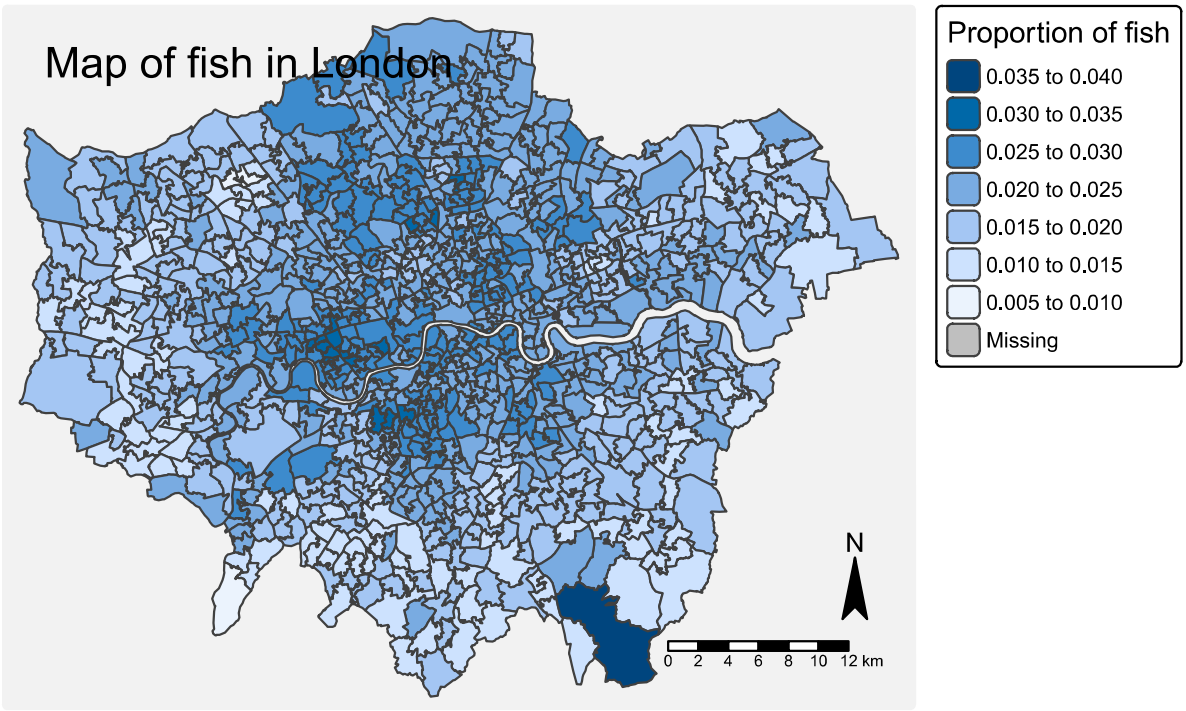

```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```



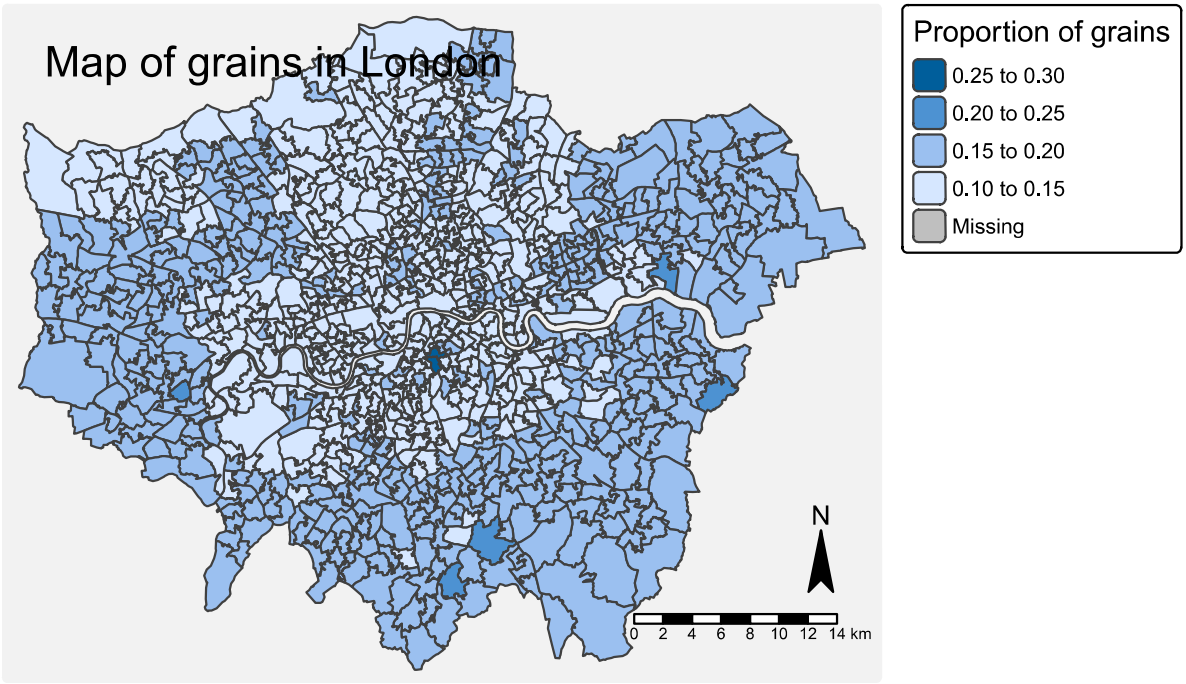

```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```



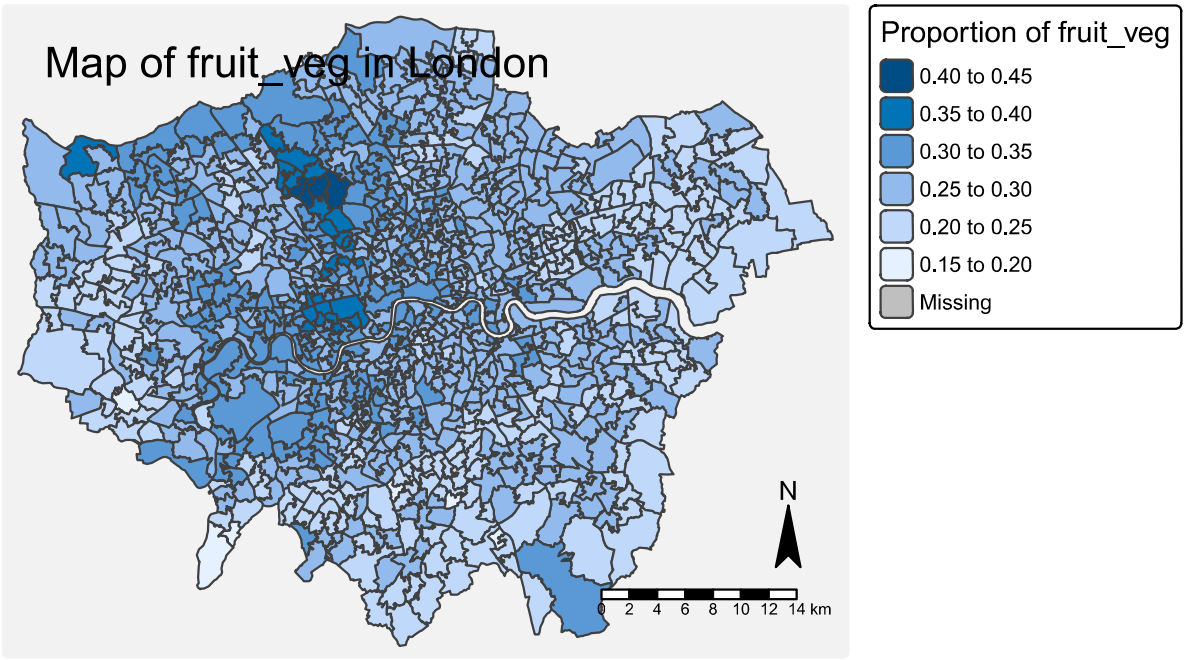

```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```



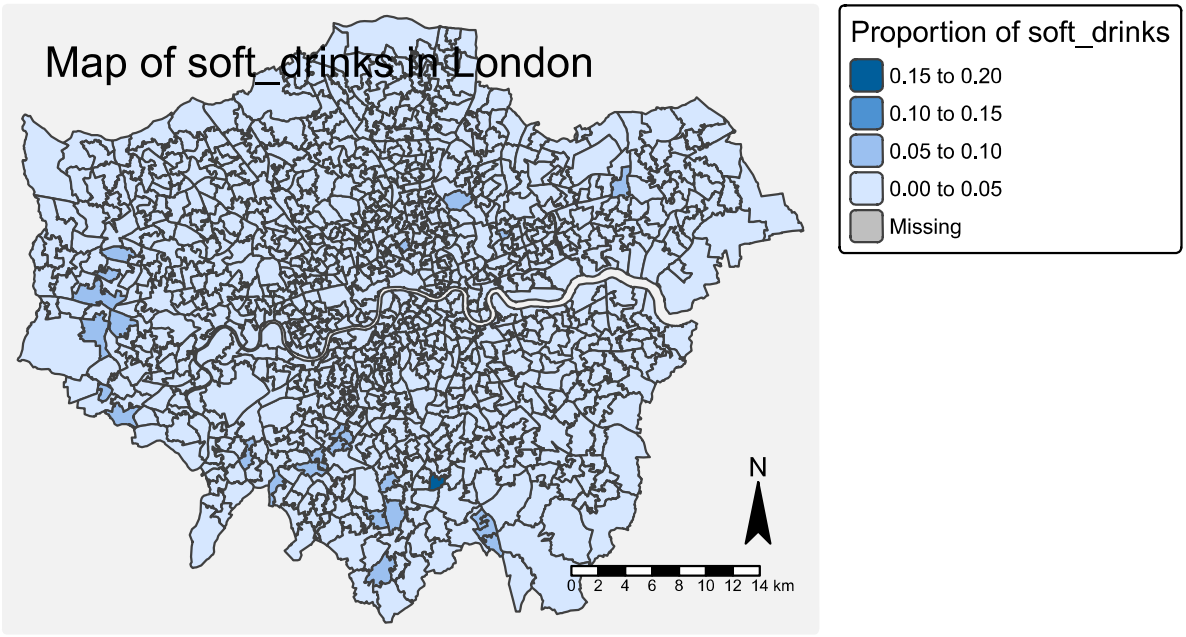

```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```



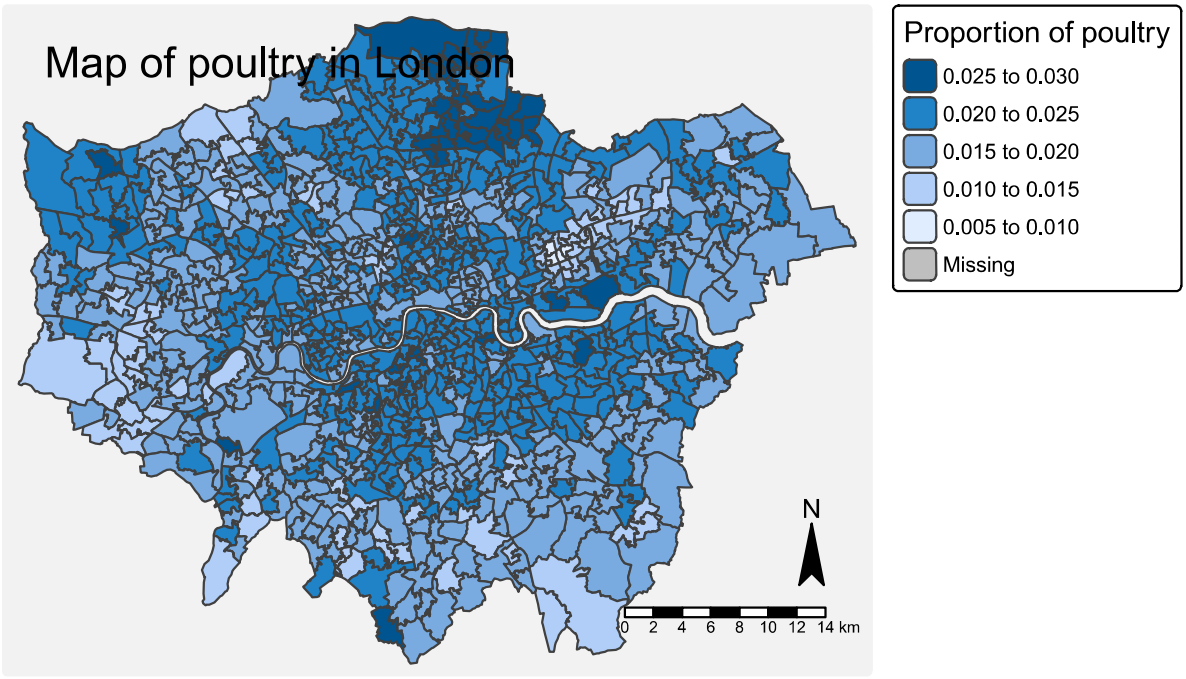

```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```



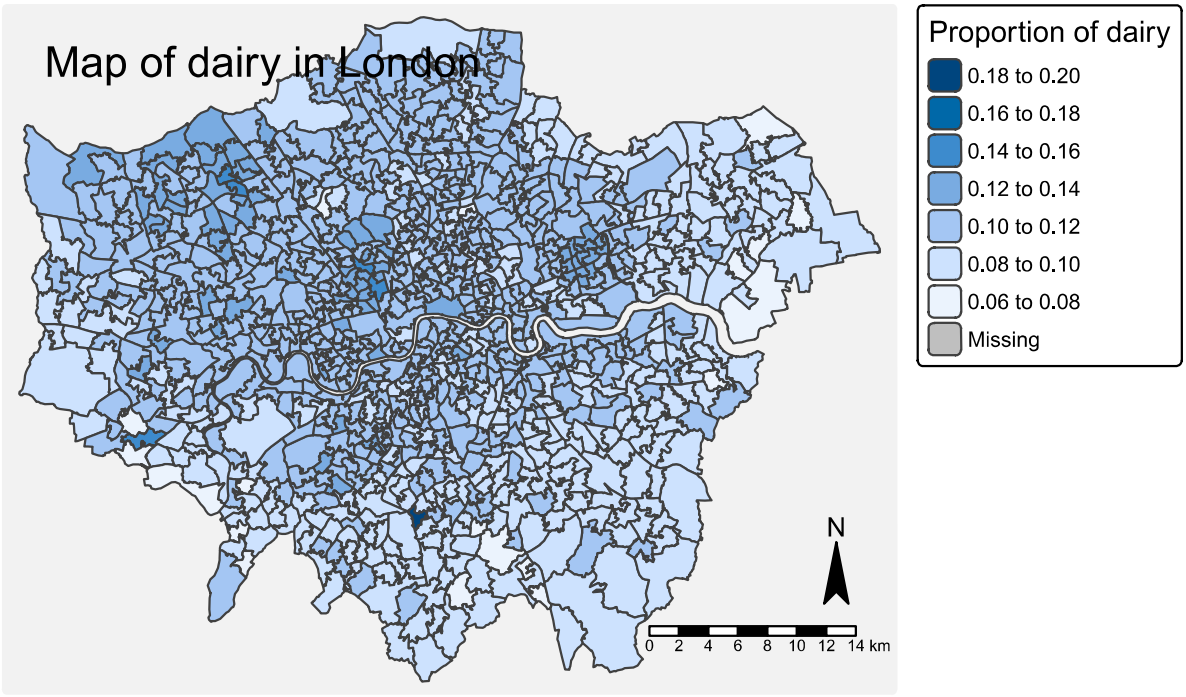

```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```

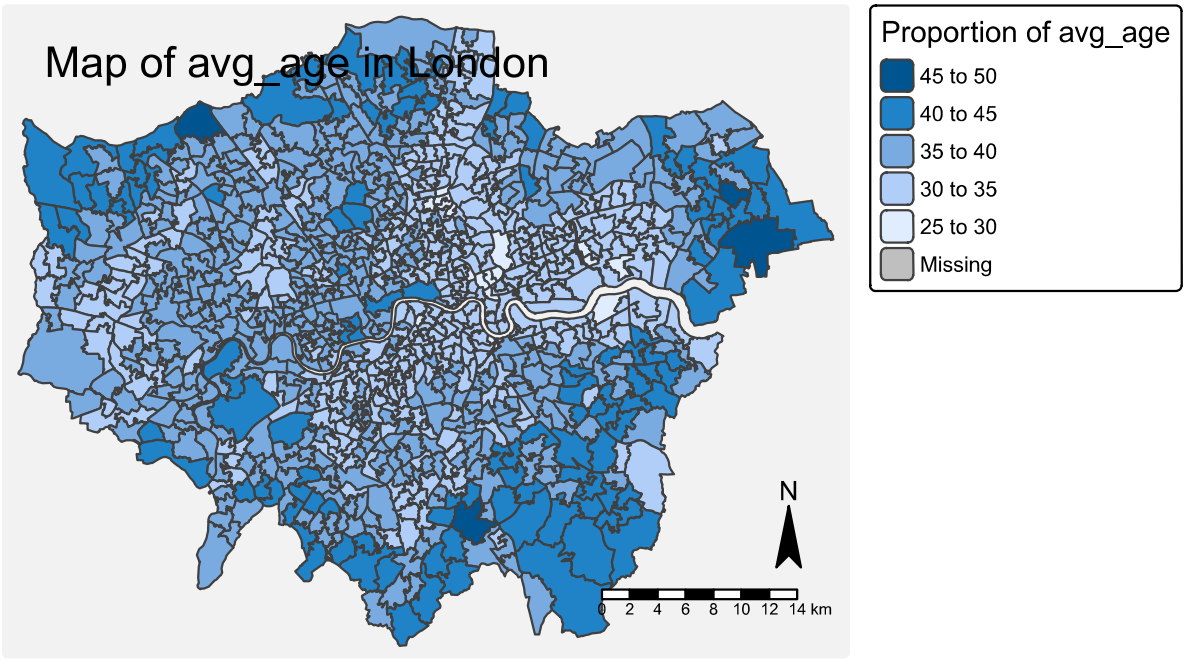



```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```




```
##
## -- tmap v3 code detected -------------------------------------------------
## [v3->v4] `tm_polygons()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title', 'legend.reverse' (rename to 'reverse')
## to 'fill.legend = tm_legend(<HERE>)'[tm_polygons()] Argument `fill.palette` unknown.[v3->v4] `tm_lay
```

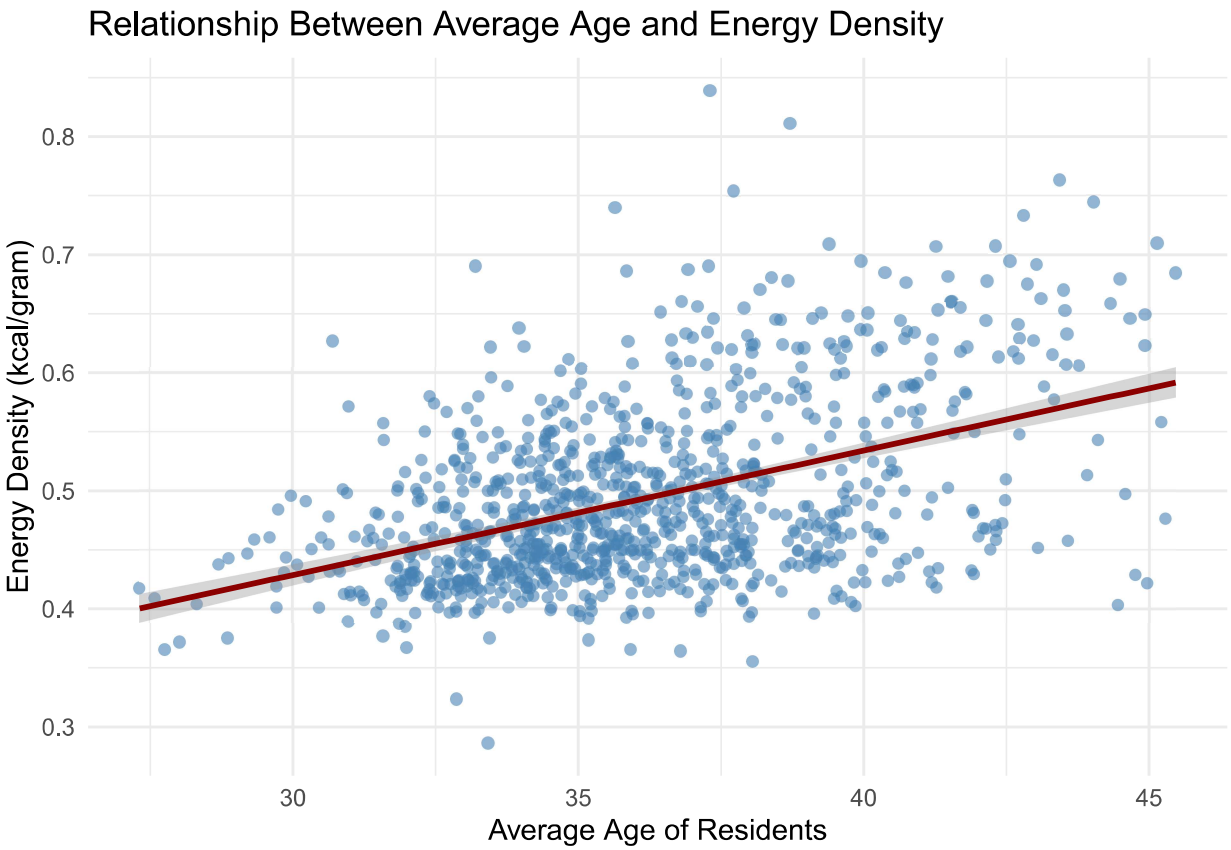

Map of dairy in London
Proportion of dairy
0.18 to 0.20
0.16 to 0.18
0.14 to 0.16
0.12 to 0.14
0.10 to 0.12
0.08 to 0.10
0.06 to 0.08
Missing
N
0 2 4 6 8 10 12 14 km



Creating graph for the demographic variable.

```
ggplot(year_msoa_grocery, aes(x = avg_age, y = energy_density)) +
  geom_point(alpha = 0.6, color = "steelblue") +
  geom_smooth(method = "lm", se = TRUE, color = "darkred") +
  labs(
    title = "Relationship Between Average Age and Energy Density",
    x = "Average Age of Residents",
    y = "Energy Density (kcal/gram)"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


Relationship Between Average Age and Energy Density
Energy Density (kcal/gram)
0.8
0.7
0.6
0.5
0.4
0.3
30
35
40
45
Average Age of Residents