

o1-Preview Writes the R Code for the Visual

```
key lat SortOrder
Coffee 47.65683325 1
Coffee 47.6620911 1
Coffee 47.66987228 1
Coffee 47.66466508 1
Coffee 47.66178131 1
Coffee 47.657406 1
Coffee 47.6587736 1
Coffee 47.66381741 1
Coffee 47.6597958 1
Coffee 47.66148 1
Coffee 47.661484 1
Coffee 47.66325 1
Coffee 47.65856199 1
Coffee 47.66407356 1
Coffee 47.65998 1
Coffee 47.6633732 1
Coffee 47.66105652 1
Coffee 47.66114 1
Coffee 47.66003 1
Coffee 47.6628435 1
Coffee 47.65538022 1
Coffee 47.66279 1
Coffee 47.661621 1
Coffee 47.65466709 1
Coffee 47.65818 1
Coffee 47.65957721 1
Coffee 47.6533553 1
Coffee 47.65851 1
Coffee 47.65637603 1
Pubs, Lounges 47.66479 2
Pubs, Lounges 47.6680599 2
Pubs, Lounges 47.65784 2
Pubs, Lounges 47.65735 2
Pubs, Lounges 47.66144279 2
Pubs, Lounges 47.66143739 2
Pubs, Lounges 47.658947 2
Pubs, Lounges 47.66171 2
Pubs, Lounges 47.667652 2
Pubs, Lounges 47.66178131 2
Pubs, Lounges 47.65553 2
Pubs, Lounges 47.66081238 2
Bubble Tea, Juice 47.66325 3
Bubble Tea, Juice 47.66105 3
Bubble Tea, Juice 47.661621 3
Bubble Tea, Juice 47.66325 3
Bubble Tea, Juice 47.65687482 3
```

Bubble Tea, Juice 47.65714 3
Bubble Tea, Juice 47.6608 3
Bubble Tea, Juice 47.6628435 3
Bubble Tea, Juice 47.66381741 3
Bubble Tea, Juice 47.664773 3
Bubble Tea, Juice 47.661484 3
Bubble Tea, Juice 47.66407356 3
Vietnamese 47.65878 4
Vietnamese 47.6584999 4
Vietnamese 47.65547 4
Vietnamese 47.6577 4
Vietnamese 47.65919 4
Vietnamese 47.65813446 4
Vietnamese 47.65822 4
Vietnamese 47.65942228 4
Thai 47.65813446 5
Thai 47.6587699 5
Thai 47.65928 5
Thai 47.6642113 5
Thai 47.65547 5
Thai 47.662455 5
Thai 47.6675671 5
Thai 47.66679 5
Thai 47.65880205 5
Chinese 47.66144 6
Chinese 47.6643 6
Chinese 47.65718842 6
Chinese 47.66174 6
Chinese 47.6594399 6
Chinese 47.662628 6
Chinese 47.65688562 6
Chinese 47.66105 6
Chinese 47.65714 6
Chinese 47.66236 6
Chinese 47.66389 6
Chinese 47.65813446 6
Chinese 47.66815 6
Chinese 47.66510105 6
Japanese 47.66671 7
Japanese 47.664349 7
Japanese 47.65809706 7
Japanese 47.6617467 7
Japanese 47.65999898 7
Japanese 47.657078 7
Japanese 47.65964 7
Japanese 47.66676083 7
Japanese 47.65799 7
Japanese 47.66359 7
Japanese 47.6577399 7
Japanese 47.65718842 7
Japanese 47.65902 7
Japanese 47.66105652 7
Korean 47.667652 8
Korean 47.6577399 8

Korean 47.65714 8
Korean 47.664349 8
Korean 47.65847 8
Korean 47.66208 8
Korean 47.65718842 8
Korean 47.6577399 8
Korean 47.66671 8
Korean 47.658201 8
Middle Eastern 47.6587699 9
Middle Eastern 47.6623521 9
Middle Eastern 47.6590045 9
Middle Eastern 47.65805 9
Middle Eastern 47.66093 9
Middle Eastern 47.66244 9
Middle Eastern 47.66295 9
Middle Eastern 47.65867 9
Middle Eastern 47.66796542 9
Middle Eastern 47.6691895 9
Middle Eastern 47.66473974 9
Middle Eastern 47.65991 9
Indian, Pakistani 47.669083 10
Indian, Pakistani 47.66473974 10
Indian, Pakistani 47.65898 10
Indian, Pakistani 47.6641 10
Indian, Pakistani 47.6587699 10
Indian, Pakistani 47.6623521 10
Pizza 47.6686016 11
Pizza 47.6558651 11
Pizza 47.658947 11
Pizza 47.667652 11
Pizza 47.66729 11
Pizza 47.65847 11
Pizza 47.662201 11
Pizza 47.65784 11
Pizza 47.66671 11
Burgers 47.65588908 12
Burgers 47.659244 12
Burgers 47.6582099 12
Burgers 47.66171 12
Burgers 47.66143739 12
Burgers 47.6616201 12
Burgers 47.66464 12
Burgers 47.6616432 12
Burgers 47.66476 12
American 47.6680599 13
American 47.657772 13
American 47.6616432 13
American 47.65651 13
American 47.6616201 13
American 47.66081238 13
Breakfast, Brunch 47.6565299 14
Breakfast, Brunch 47.65851 14
Breakfast, Brunch 47.65651 14
Breakfast, Brunch 47.66295 14

Breakfast, Brunch 47.657772 14
 Breakfast, Brunch 47.6597958 14
 Breakfast, Brunch 47.66466508 14
 Breakfast, Brunch 47.6595172 14
 Bakeries 47.66178131 15
 Bakeries 47.65998 15
 Bakeries 47.6633732 15
 Bakeries 47.6582099 15
 Bakeries 47.66105 15
 Bakeries 47.66887 15
 Seafood 47.66015 16
 Seafood 47.66464 16
 Seafood 47.6616432 16
 Hawaiian 47.66359 17
 Hawaiian 47.664349 17
 Hawaiian 47.6627388 17
 Vegetarian, Vegan 47.6675671 18
 Vegetarian, Vegan 47.664773 18
 Vegetarian, Vegan 47.6686016 18
 Vegetarian, Vegan 47.66838 18
 Mexican, Latin American 47.65714 19
 Mexican, Latin American 47.6659099 19
 Mexican, Latin American 47.65923 19
 Mexican, Latin American 47.65161 19
 Mexican, Latin American 47.6643199 19
 Mexican, Latin American 47.6595391 19
 Mexican, Latin American 47.66146732 19
 Mexican, Latin American 47.66475922 19

Create an R script for Power BI's R visual to reproduce a ridgeline density plot. The ridgeline plot should represent various food categories along different street sections, as seen in the original chart. The input dataset includes three columns: key (food categories), lat (latitude coordinates of street segments), and SortOrder (ordering of the categories on the Y-axis). Ensure the visual clearly distinguishes the different categories using different colors and provide smoothed density curves across each category.

Here are the details:

- The X-axis should represent the street segments, mapped by latitude (lat).
- The Y-axis should represent the different food categories (provided by key).
- Sort the Y-axis according to the SortOrder column.
- Use different colors for each category to distinguish them.
- Add a title to the plot: "A Mile-Long Global Food Market: Mapping Cuisine from 'The Ave'".
- Set individual density ridges for each food category using a smooth kernel density estimate.
- Ensure that overlapping is visualized in a way that density peaks are visible for each category.
- Assign a fixed scale for the X-axis and group the curves by food categories to maintain the overall layout seen in the provided image.
- Avoid background grids and include minimal axis lines to keep the focus on the density curves.

Generating the script

I'm tasked with creating an R script for Power BI's R visual to produce a ridgeline density plot, representing food categories along various street sections from the provided data.

Establishing the framework

Mapping out the script to plot street segments against food categories, ensuring clarity with distinct colors, concise titles, and smooth density curves. Progressing towards a visually engaging global food market representation.

Crafting the script

I'm detailing an R script to generate a ridgeline density plot using the 'ggridges' package. The script will read data from the 'dataset' dataframe, containing 'key', 'lat', and 'SortOrder' columns, and create the plot with ggplot2.

Preparing the environment

I'm pulling together the necessary R libraries for Power BI's R visual, focusing on ggplot2 and ggridges, followed by ensuring the data is properly sorted.

Crafting the plot

I'm assembling a ggplot for ridgeline plots, adjusting themes and axis appearances, and setting vertical scaling to visualize overlapping density peaks of food categories.

Tweaking plot parameters

I'm thinking through adjusting 'scale' in 'geom_density_ridges' and setting a fixed X-axis for alignment. Using specific colors for each category seems like a good move to separate them clearly.

Crafting the plot

I'm assembling the components for a ridgeline plot in R, using ggplot2 and ggridges. This involves setting aesthetics, adjusting theme elements, and ensuring the 'key' variable is ordered by 'SortOrder'.

Ensuring order

I'm working through ways to order 'key' by 'SortOrder' in R. Using 'reorder(dataset\$key, dataset\$SortOrder)' seems effective, as it maintains 'key' values' relative order.

Refining the process

I'm setting the levels of 'key' in a dataset to ensure they match the unique 'key' values ordered by 'SortOrder'. This involves sorting and using unique 'key' values to define the factor levels.

Enhancing the script

I'm refining the ggplot2 script for visualizing datasets. Upgraded 'scale_fill_categorical()' to 'scale_fill_brewer(palette = "Set3")' or 'scale_fill_viridis_d()' for discrete variables to boost clarity and effectiveness.

Mapping out options

I'm assigning a fixed X-axis scale in `ggplot` to ensure clarity in the density ridges plot, removing background grids, and tweaking axis lines to enhance the focus on density curves.

Tweaking the visualization

I'm adjusting the 'scale' parameter to refine ridge overlap. Next, I'll tweak 'alpha' for exit density transparency, aiming for clear peaks and minimal grid noise.

Adjusting the approach

I will include 'alpha = 0.8' in 'geom_density_ridges' or 'aes'. This should address the issue. The default behavior is also considered acceptable.

Sure! Below is an R script designed for Power BI's R visual to create the specified ridgeline density plot. This script assumes that your dataset includes three columns: `key` (food categories), `lat` (latitude coordinates), and `SortOrder` (ordering of the categories on the Y-axis).

The script uses the `ggplot2` and `ggridges` packages to generate the plot, and it ensures that the different food categories are distinguished by colors, sorted according to `SortOrder`, and that the density curves are smoothed and overlapping appropriately.

R

```
# Load necessary libraries
library(ggplot2)
library(ggribes)
library(viridis) # For color scales

# Ensure 'key' is a factor ordered by 'SortOrder'
key_levels <- unique(dataset[order(dataset$SortOrder), "key"])
dataset$key <- factor(dataset$key, levels = key_levels)

# Create the ridgeline density plot
ggplot(dataset, aes(x = lat, y = key, fill = key)) +
  geom_density_ridges(
    scale = 1.5,          # Adjusts overlap between ridgelines
    rel_min_height = 0.01, # Removes outline around ridgelines
    color = NA,           # Slight transparency to see overlaps
    alpha = 0.8
  ) +
  coord_cartesian(xlim = range(dataset$lat)) + # Fixed X-axis scale
  theme_ridges(center_axis_labels = TRUE) +
  theme(
    axis.title.y = element_blank(), # Removes Y-axis title
    axis.title.x = element_blank(), # Removes X-axis title
    panel.grid = element_blank(),   # Removes background grids
    axis.ticks = element_line(color = "grey50"),
    axis.line = element_line(color = "grey50"),
    legend.position = "none"        # Hides legend
  ) +
  labs(title = "A Mile-Long Global Food Market: Mapping Cuisine from 'The Ave'") +
  scale_fill_viridis_d() # Distinct colors for each category
```

Explanation of the Script:

- **Libraries:**
 - `ggplot2` for plotting.
 - `ggribes` for creating the ridgeline plots.
 - `viridis` for an appealing and colorblind-friendly palette.
- **Data Preparation:**
 - The `key` column is converted to a factor with levels ordered by `SortOrder` to control the Y-axis order.
- **Plot Construction:**
 - `ggplot()` initializes the plot with `lat` on the X-axis and the ordered `key` on the Y-axis.
 - `geom_density_ridges()` creates the ridgeline plots with adjusted overlap (`scale`), minimal height (`rel_min_height`), no outline (`color = NA`), and slight transparency (`alpha = 0.8`).
 - `coord_cartesian()` ensures a fixed scale for the X-axis based on the range of `lat`.
 - `theme_ridges()` along with additional `theme()` settings removes background grids and axis titles to keep the focus on the density curves.
 - `labs()` adds the specified title to the plot.
 - `scale_fill_viridis_d()` applies distinct colors to each food category.

Note: Make sure that the `ggplot2`, `ggribes`, and `viridis` packages are installed in your Power BI environment. If not, you can install them using the `install.packages()` function, but remember that Power BI's R environment might have restrictions on package installations.

Sample Visualization:

The script will produce a ridgeline plot where each food category is represented on the Y-axis, sorted according to the `SortOrder`. The X-axis represents the latitude (`lat`), effectively mapping the street segments. Each category's density curve is smoothed and overlaps slightly with others, highlighting the density peaks. The use of different colors for each category enhances the visual distinction between them.

By running this script in Power BI's R visual with your dataset, you should obtain the desired ridgeline density plot that meets all the specified requirements.

