EDA

```
transaction <- arrow::read_parquet(file.path(params$path, "transaction.parquet"))
store <- arrow::read_parquet(file.path(params$path, "store.parquet"))
product <- arrow::read_parquet(file.path(params$path, "product.parquet"))
```

Summaries

```
describe <- skimr::skim_with(
    numeric = skimr::sfl(
        mean = ~ round(mean(., na.rm = TRUE), 2),
        sd = ~ round(sd(., na.rm = TRUE), 2),
        min = ~ round(min(., na.rm = TRUE), 2),
        max = ~ round(max(., na.rm = TRUE), 2),
        p0 = NULL, p50 = NULL, p100 = NULL, hist = NULL
    ),
    append = TRUE
)
```

Transactions

```
describe(tibble::as_tibble(transaction) |> dplyr::select(-week))
```

Table 1: Data summary

Name	dplyr::select(tibble::as_…
Number of rows	524950
Number of columns	12

Column type frequency:	
character	2
numeric	9
POSIXct	1
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
store_id	0	1	3	5	0	77	0
upc_id	0	1	10	11	0	55	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p25	p75	min	max
units	0	1	19.61	29.93	4.00	24.00	0.00	1800.00
visits	0	1	17.17	24.74	4.00	21.00	1.00	1340.00
hhs	0	1	16.78	24.19	4.00	21.00	1.00	1286.00
spend	0	1	53.20	68.18	13.36	67.60	0.00	2952.00
price	23	1	3.38	1.56	2.36	4.49	0.00	11.46
base_price	185	1	3.60	1.63	2.50	4.59	0.55	11.46
feature	0	1	0.08	0.28	0.00	0.00	0.00	1.00
display	0	1	0.11	0.31	0.00	0.00	0.00	1.00
tpr_only	0	1	0.13	0.34	0.00	0.00	0.00	1.00

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
week_end_date	0	1	2009-01-14	2012-01-04	2010-07-21	156

Stores

```
describe(store)
```

Table 5: Data summary

Name	store
Number of rows	77
Number of columns	9
Column type frequency:	
character	5
numeric	4
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
store_id	0	1	3	5	0	77	0
store_name	0	1	4	20	0	73	0
address_city_name	0	1	4	15	0	51	0
address_state_prov_code	0	1	2	2	0	4	0
seg_value_name	0	1	5	18	0	4	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p25	p75	min	max
msa_code	0	1.00	21291.69	6208.98	17140.0	26420.00	13140.00	47540.00
parking_space_qty	51	0.34	479.27	411.60	276.0	452.50	17.00	1859.00
sales_area_size_num	0	1.00	48922.40	13614.21	42437.0	54448.00	10788.00	86517.00
avg_weekly_baskets	0	1.00	24209.24	8806.79	17009.9	29386.42	10434.71	54052.52

Products

```
describe(product)
```

Table 8: Data summary

Name	product
Number of rows	58

Number of columns	8
Column type frequency:	
character	6
numeric	2
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
upc_id	0	1	10	11	0	58	0
description	0	1	11	25	0	55	0
manufacturer	0	1	4	13	0	17	0
category	0	1	10	21	0	4	0
sub_category	0	1	8	27	0	7	0
product_size	0	1	4	8	0	31	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p25	p75	min	max
volume_ml	47	0.19	781.45	336.41	516.0	1000.00	500.00	1500.0
mass_oz	11	0.81	17.85	7.51	13.1	21.35	0.11	32.7

Outliers

```
#' Outliers
#'
#' Print total number of outliers in transaction.
#' @param transaction A `tsibble` containing the transaction dataset.
#' @returns A `tibble` containing the outliers.
outliers <- function(transaction) {
    is_outlier <- function(x) {
        return(
            (x < quantile(x, 0.25, na.rm = TRUE) - 3 * IQR(x, na.rm = TRUE)) |
                (x > quantile(x, 0.75, na.rm = TRUE) + 3 * IQR(x, na.rm = TRUE))
```

```
        )
    }

    outliers <- transaction |>
        tibble::as_tibble() |>
        dplyr::reframe(
            across(
                .cols = c(units, visits, hhs, spend, price, base_price),
                .fns = is_outlier,
                .names = "outlier_{.col}"
            )
        ) |>
        dplyr::mutate(
            week = transaction$week,
            store_id = transaction$store_id,
            upc_id = transaction$upc_id,
            .before = 1
        )

    # Number of outliers per feature
    return(
        outliers |>
            dplyr::select(tidyselect::contains("outlier")) |>
            dplyr::reframe(dplyr::across(tidyselect::everything(), sum))
    )
}

outliers(transaction)
```

outlier_units	outlier_visits	outlier_hhs	outlier_spend	outlier_price	outlier_base_price
15783	16156	15532	12555	NA	NA

What is the range of prices offered on product?

```
#' Price range
#'
#' Print the range of prices offered on product.
#' @param transaction A `tsibble` containing the transaction data.
```

```r
#' @returns A `skim` summary.
prices_range <- function(transaction) {
    # Skim summary custom function
    price_summary <- skimr::skim_with(
        numeric = skimr::sfl(
            min = ~ min(., na.rm = TRUE),
            max = ~ max(., na.rm = TRUE),
            mean = ~ mean(., na.rm = TRUE),
            sd = ~ sd(., na.rm = TRUE)
        ),
        append = FALSE
    )

    prices_range <- transaction |>
        tibble::as_tibble() |>
        dplyr::select(upc_id, price) |>
        dplyr::group_by(upc_id) |>
        price_summary()
    return(prices_range)
}

prices_range(transaction)
```

Table 12: Data summary

Name	dplyr::group_by(…)
Number of rows	524950
Number of columns	2
_____	
Column type frequency:	
numeric	1
_____	
Group variables	upc_id

Variable type: numeric

skim_variable	upc_id	n_missing	complete_rate	min	max	mean	sd
price	1111009477	0	1	0.89	1.83	1.30	0.20
price	1111009497	0	1	0.86	1.69	1.30	0.20
price	1111009507	0	1	0.80	1.69	1.31	0.20

skim_variable	upc_id	n_missing	complete_rate	min	max	mean	sd
price	1111035398	1	1	1.00	4.69	3.15	0.49
price	1111038078	1	1	0.47	3.08	1.45	0.42
price	1111038080	1	1	0.46	4.18	1.45	0.42
price	1111085319	0	1	1.07	1.99	1.76	0.13
price	1111085345	0	1	0.93	2.00	1.76	0.12
price	1111085350	0	1	1.03	2.49	2.14	0.21
price	1111087395	0	1	1.40	5.29	3.59	0.54
price	1111087396	1	1	0.01	5.49	3.58	0.53
price	1111087398	0	1	1.78	5.29	3.60	0.53
price	1600027527	0	1	1.50	3.39	2.77	0.37
price	1600027528	0	1	1.50	5.19	4.27	0.56
price	1600027564	0	1	0.58	3.56	2.76	0.42
price	2066200530	1	1	2.99	6.99	6.02	0.50
price	2066200531	0	1	2.99	6.99	6.01	0.52
price	2066200532	0	1	4.00	6.99	6.03	0.54
price	2840002333	1	1	1.65	3.29	2.89	0.27
price	2840004768	0	1	1.65	3.29	2.89	0.27
price	2840004770	0	1	1.65	3.29	2.88	0.27
price	3000006340	0	1	0.91	7.00	2.74	0.53
price	3000006560	0	1	1.06	3.09	2.49	0.30
price	3000006610	2	1	1.00	3.09	2.48	0.32
price	31254742725	0	1	1.00	5.14	4.05	0.57
price	31254742735	2	1	1.50	6.19	4.71	0.52
price	31254742835	2	1	1.99	7.59	4.71	0.52
price	3500068914	0	1	0.50	7.99	7.50	1.22
price	3700019521	2	1	0.81	11.46	3.00	0.53
price	3700031613	0	1	1.79	5.29	4.06	0.46
price	3700044982	2	1	2.50	6.91	4.99	0.45
price	3800031829	0	1	1.54	4.20	3.16	0.38
price	3800031838	0	1	0.85	3.86	2.89	0.38
price	3800039118	0	1	1.45	3.77	2.93	0.43
price	4116709428	1	1	1.75	4.99	4.55	0.29
price	4116709448	0	1	2.29	5.19	4.56	0.30
price	4116709565	1	1	1.00	4.99	4.80	0.34
price	7027312504	0	1	0.66	3.49	1.83	0.35
price	7027316204	0	1	0.66	3.49	1.76	0.34
price	7027316404	0	1	0.00	3.49	1.76	0.33
price	7110410455	0	1	1.61	2.79	2.27	0.24
price	7110410470	0	1	1.95	2.79	2.27	0.24
price	7110410471	0	1	1.80	2.79	2.26	0.24
price	7192100336	0	1	4.00	7.49	5.97	0.84

skim_variable	upc_id	n_missing	complete_rate	min	max	mean	sd
price	7192100337	0	1	4.19	7.49	5.98	0.84
price	7192100339	1	1	4.17	7.49	5.98	0.84
price	7218063052	1	1	3.94	8.91	6.31	0.98
price	7218063979	1	1	1.49	7.89	6.30	0.99
price	7218063983	1	1	3.99	7.89	6.30	1.00
price	7797502248	0	1	1.00	2.72	2.34	0.25
price	7797508004	0	1	2.06	3.29	2.84	0.29
price	7797508006	1	1	1.99	3.29	2.84	0.29
price	88491201426	0	1	0.84	4.99	3.19	0.25
price	88491201427	0	1	1.39	3.73	3.22	0.28
price	88491212971	0	1	1.20	3.63	2.56	0.35

What is the impact on sales of promotions, displays, or being featured in the circular?

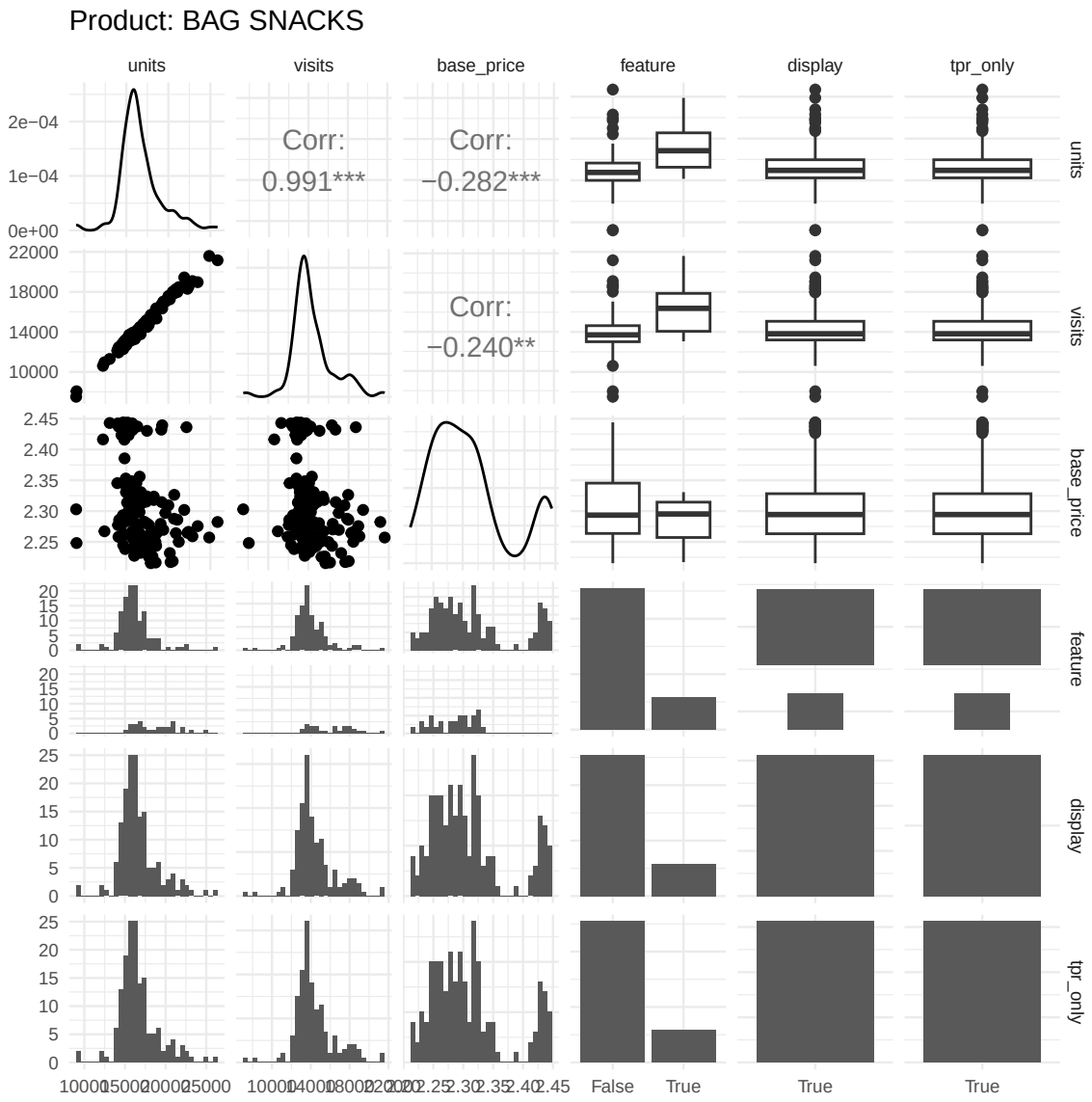
```
#' Product overview
#'
#' Helper functions to make visualizations
#' @param ts A `tibble` containing the transaction data.
#' @param selected_cat A `character` containing the selected product category.
#' @returns A `ggplotly` object.
product_overview <- function(ts, selected_cat) {
    return(
        ts |>
            dplyr::filter(category == selected_cat) |>
            tidyr::pivot_longer(
                c(units, base_price, feature, display, tpr_only),
                names_to = "col",
                values_to = "value"
            ) |>
            ggplot2::ggplot(ggplot2::aes(x = as.Date(tsibble::yearweek(week)), y = value, col
            ggplot2::geom_line() +
            ggplot2::facet_wrap(~col, scales = "free_y", ncol = 1) +
            ggplot2::scale_fill_brewer(palette = "Set1") +
            ggplot2::labs(
                title = glue::glue("Product: {selected_cat}"),
                x = "Week"
            ) +
```

```
            ggplot2::theme(
                legend.position = "none",
                axis.text.x = ggplot2::element_text(angle = 45, hjust = 1)
            )
    )
}

#' Product correlation
#'
#' Plot a pairplot to see if there is any correlation among the features.
#' @param ts A `tibble` containing the transaction data.
#' @param selected_cat A `character` containing the selected product category.
#' @returns A `ggpairs` object.
product_corr <- function(ts, selected_cat) {
    return(
        ts |>
            dplyr::filter(category == selected_cat) |>
            dplyr::mutate(
                feature = ifelse(feature == 1, "True", "False"),
                display = ifelse(display == 1, "True", "False"),
                tpr_only = ifelse(tpr_only == 1, "True", "False")
            ) |>
            dplyr::select(
                -category,
                -week
            ) |>
            GGally::ggpairs() +
            ggplot2::labs(title = glue::glue("Product: {selected_cat}")) +
            ggplot2::theme_minimal(base_size = 9)
    )
}

# I will aggregate the data by product category, in order to
# compare promotions, displays and being featured against sales.
# Aggregation will lost information but the idea is to obtain an overview.
ts <- transaction |>
    dplyr::left_join(product, by = dplyr::join_by(upc_id)) |>
    tibble::as_tibble() |>
    dplyr::group_by(week, category) |>
    dplyr::summarise(
        .groups = "drop",
        units = sum(units, na.rm = TRUE),
```

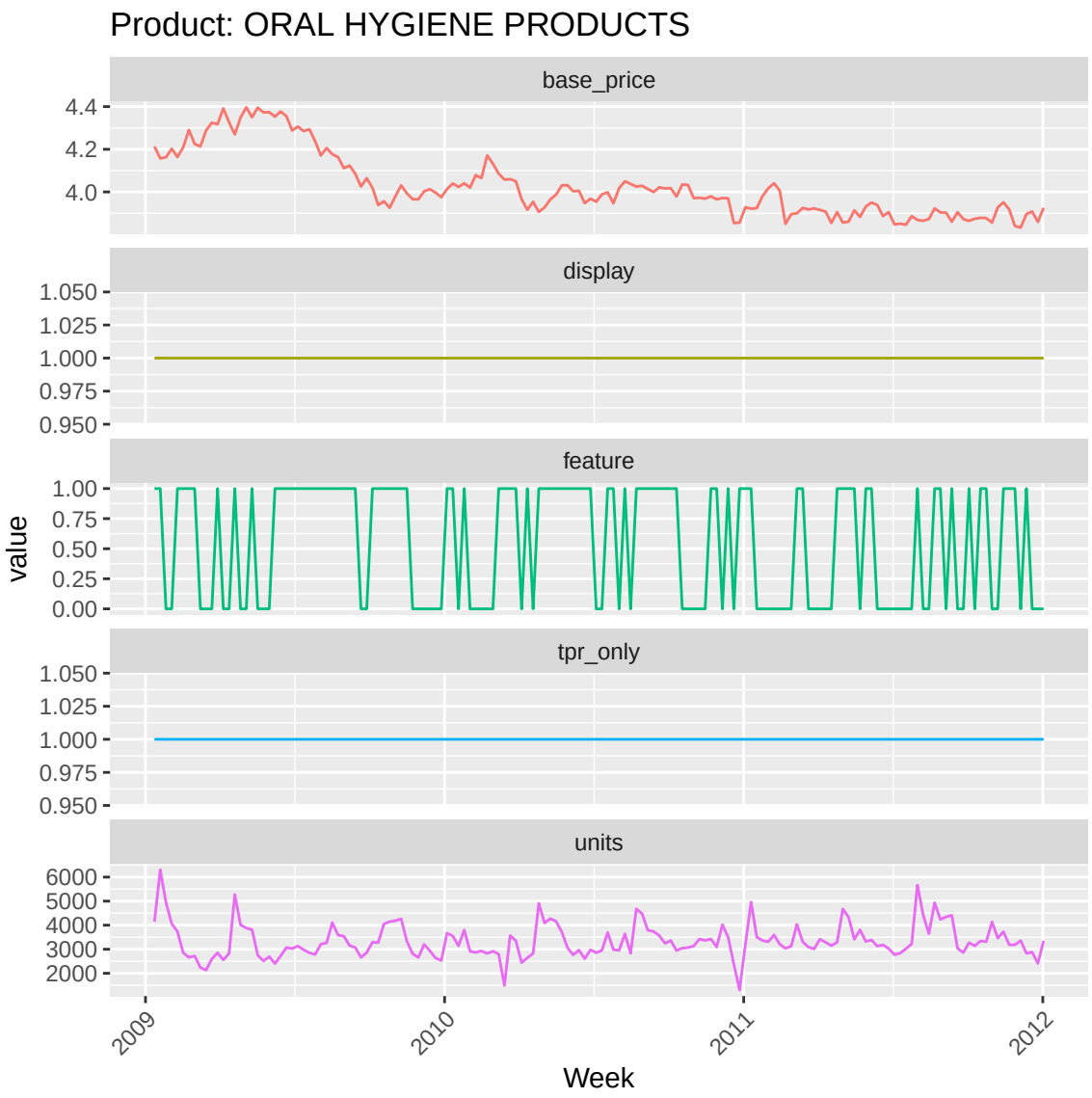
```
        visits = sum(visits, na.rm = TRUE),
        base_price = mean(base_price, na.rm = TRUE),
        feature = ceiling(mean(feature, na.rm = TRUE)),
        display = ceiling(mean(display, na.rm = TRUE)),
        tpr_only = ceiling(mean(tpr_only, na.rm = TRUE))
    )
```

```
product_overview(ts, "BAG SNACKS") |> render_plot()
```

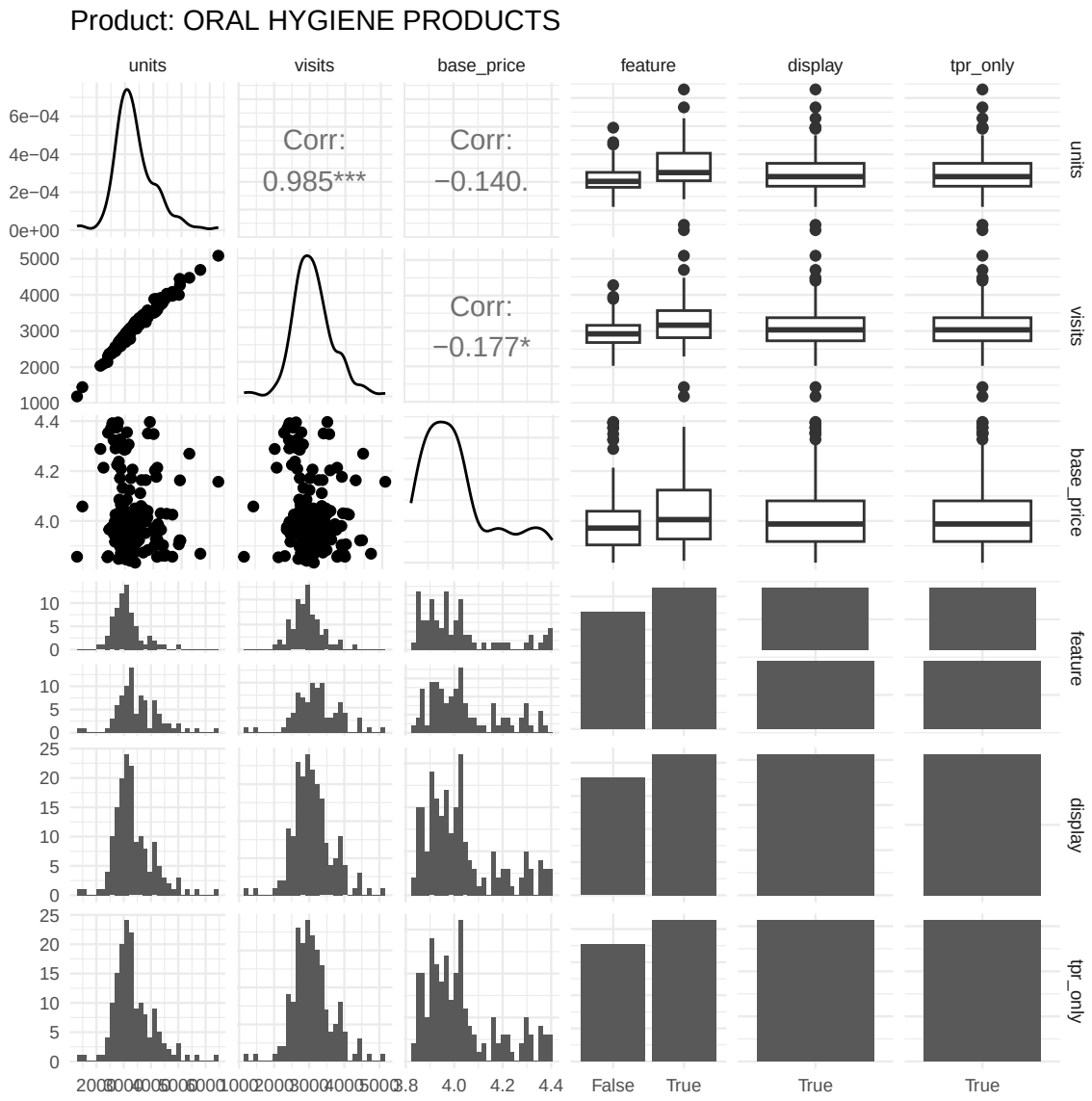
```
product_corr(ts, "BAG SNACKS") |> render_plot()
```



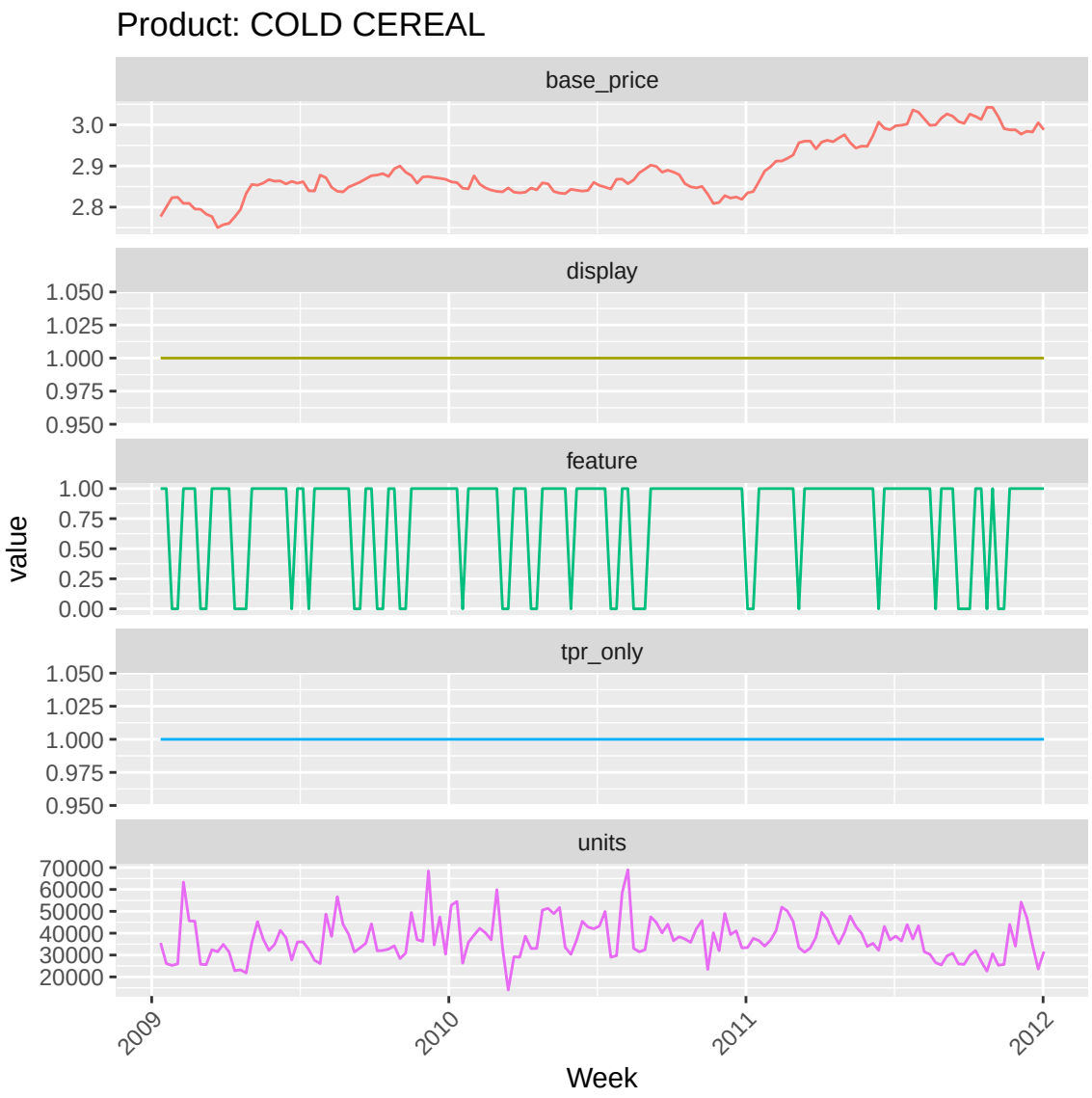
```
product_overview(ts, "ORAL HYGIENE PRODUCTS") |> render_plot()
```



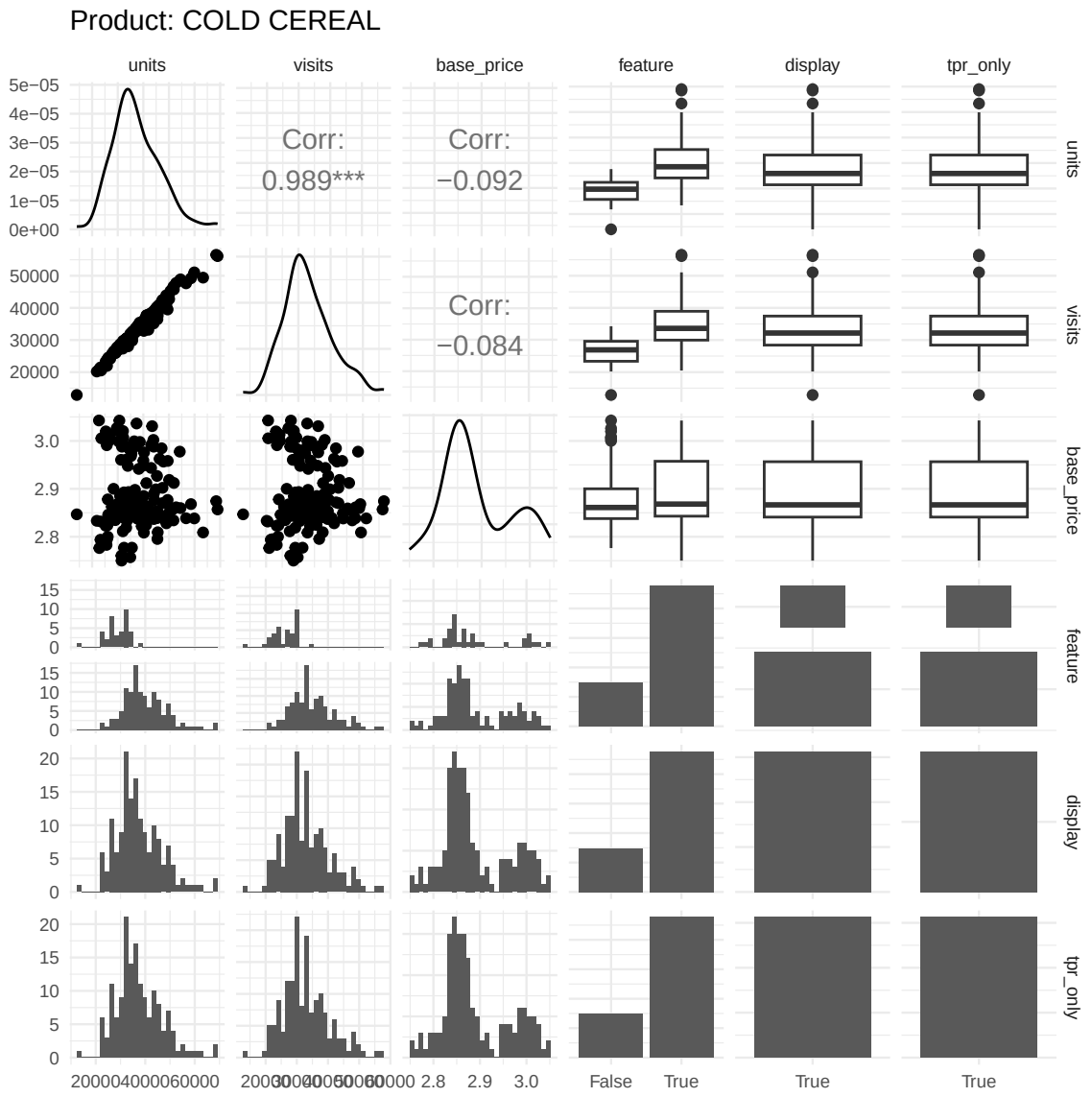
```
product_corr(ts, "ORAL HYGIENE PRODUCTS") |> render_plot()
```



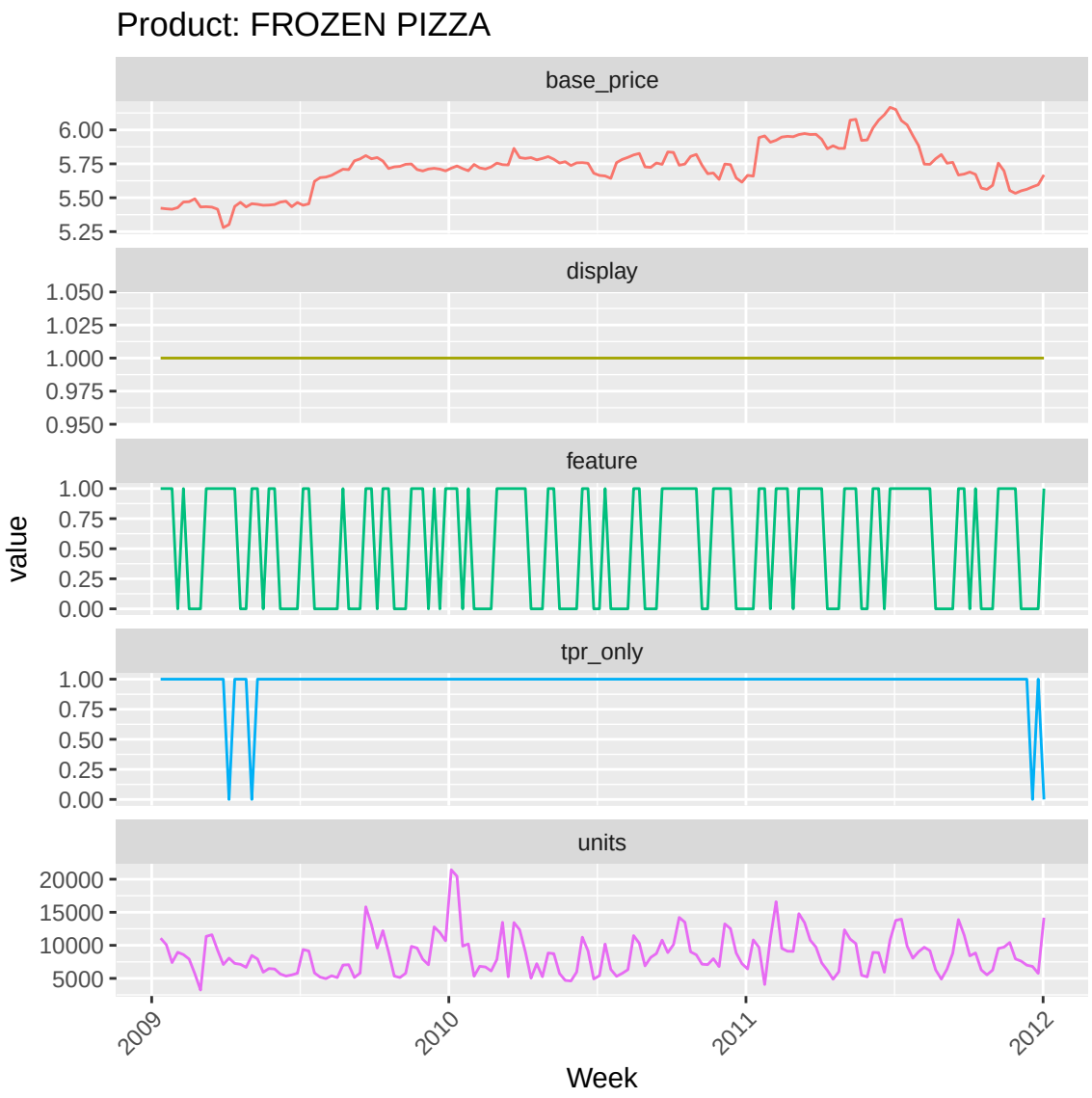
```
product_overview(ts, "COLD CEREAL") |> render_plot()
```



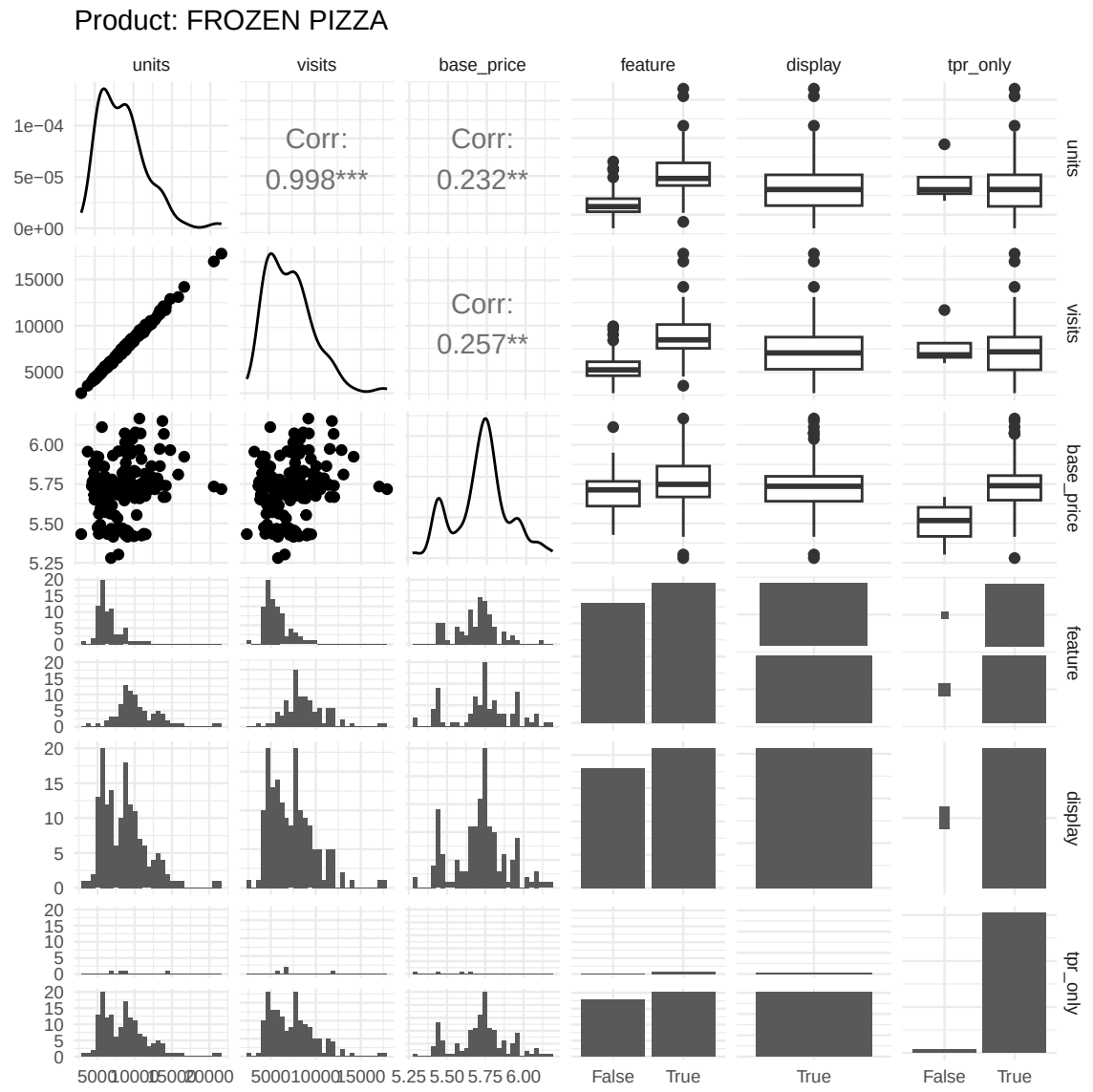
```
product_corr(ts, "COLD CEREAL") |> render_plot()
```



```
product_overview(ts, "FROZEN PIZZA") |> render_plot()
```



```
product_corr(ts, "FROZEN PIZZA") |> render_plot()
```

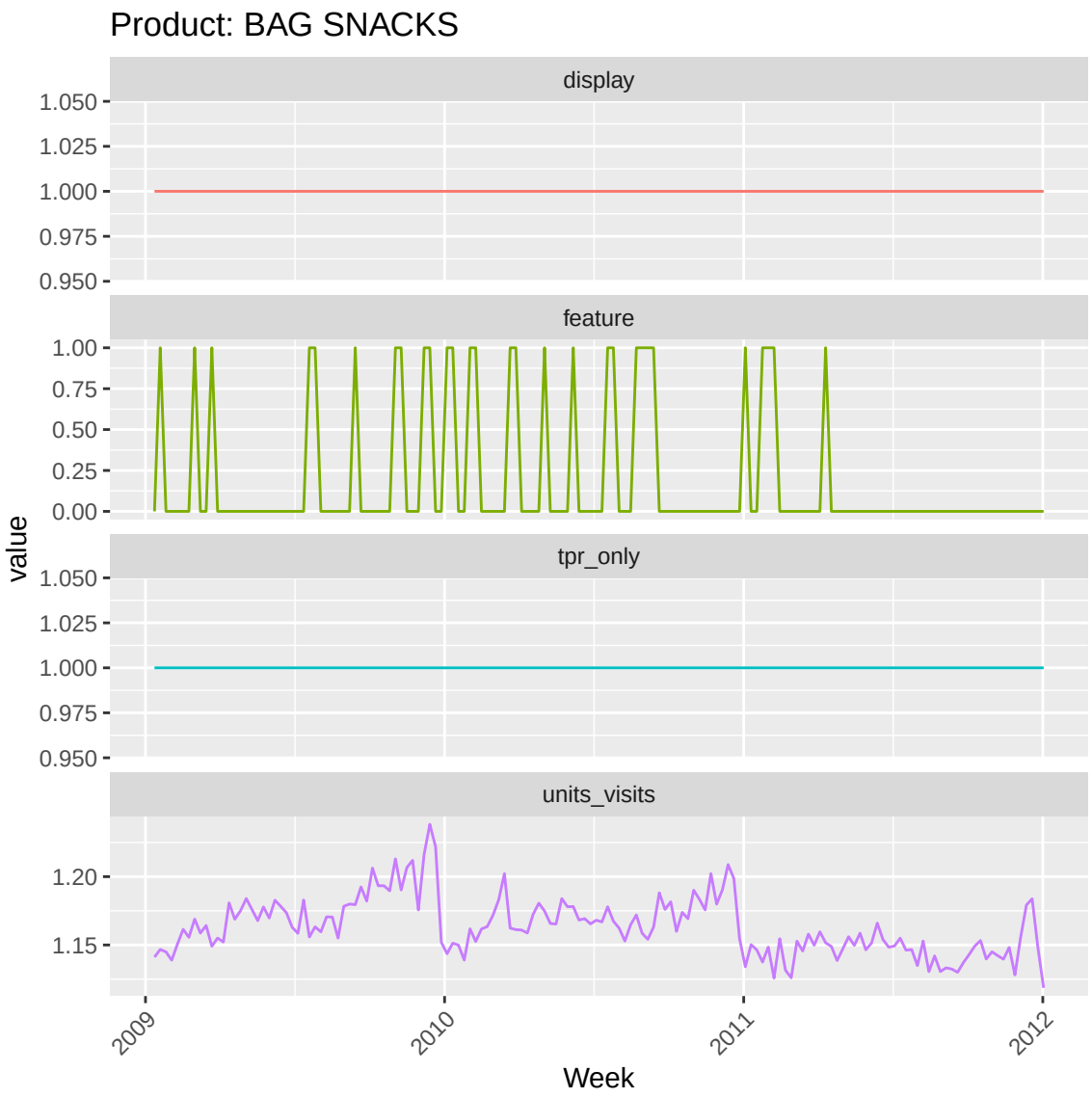


Observing the plots, we can see that exist correlation between units and promotions. This is expected, since promotions are done to increase sales. But, some promoted product don't show increase in sales. Further analysis is recommended for those product. Also, is visible that units is slightly correlated with base_price, since base_price represents the baseline price of the final price of the product.

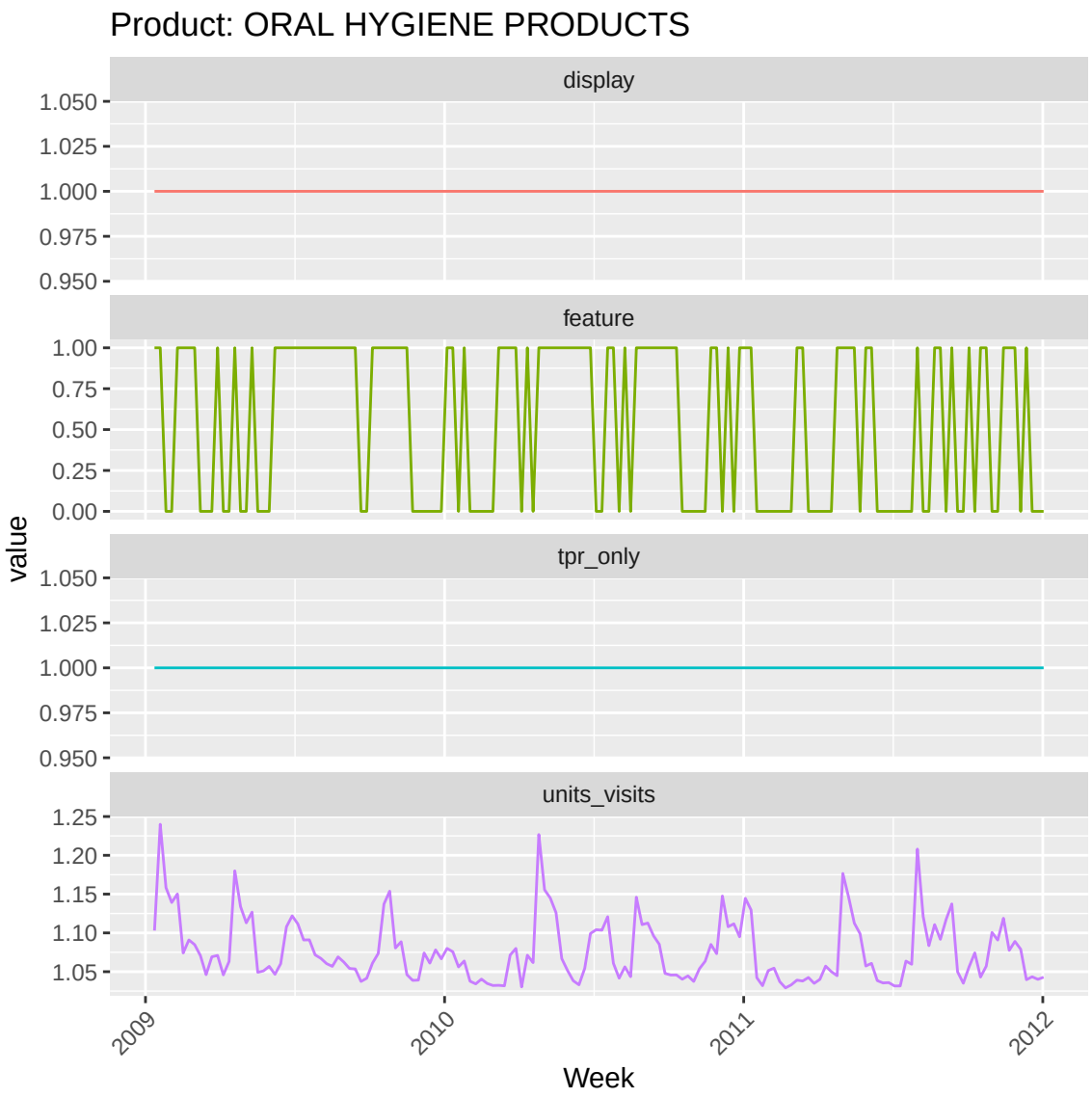
What is the impact on units/visits of promotions?

```r
#' Units/Visits promotion
#'
#' Helper functions to make visualizations.
#' @param ts A `tibble` of the transaction data.
#' @param selected_cat A `character` containing the product category to be selected.
#' @returns A `ggplotly` object.
units_visits_prom <- function(ts, selected_cat) {
    return(
        ts |>
            dplyr::filter(category == selected_cat) |>
            dplyr::mutate(units_visits = units / visits) |>
            tidyr::pivot_longer(
                c(units_visits, feature, display, tpr_only),
                names_to = "col",
                values_to = "value"
            ) |>
            ggplot2::ggplot(ggplot2::aes(x = as.Date(tsibble::yearweek(week)), y = value, col
            ggplot2::geom_line() +
            ggplot2::facet_wrap(~col, scales = "free_y", ncol = 1) +
            ggplot2::scale_fill_brewer(palette = "Set1") +
            ggplot2::labs(
                title = glue::glue("Product: {selected_cat}"),
                x = "Week"
            ) +
            ggplot2::theme(
                legend.position = "none",
                axis.text.x = ggplot2::element_text(angle = 45, hjust = 1)
            )
    )
}
```

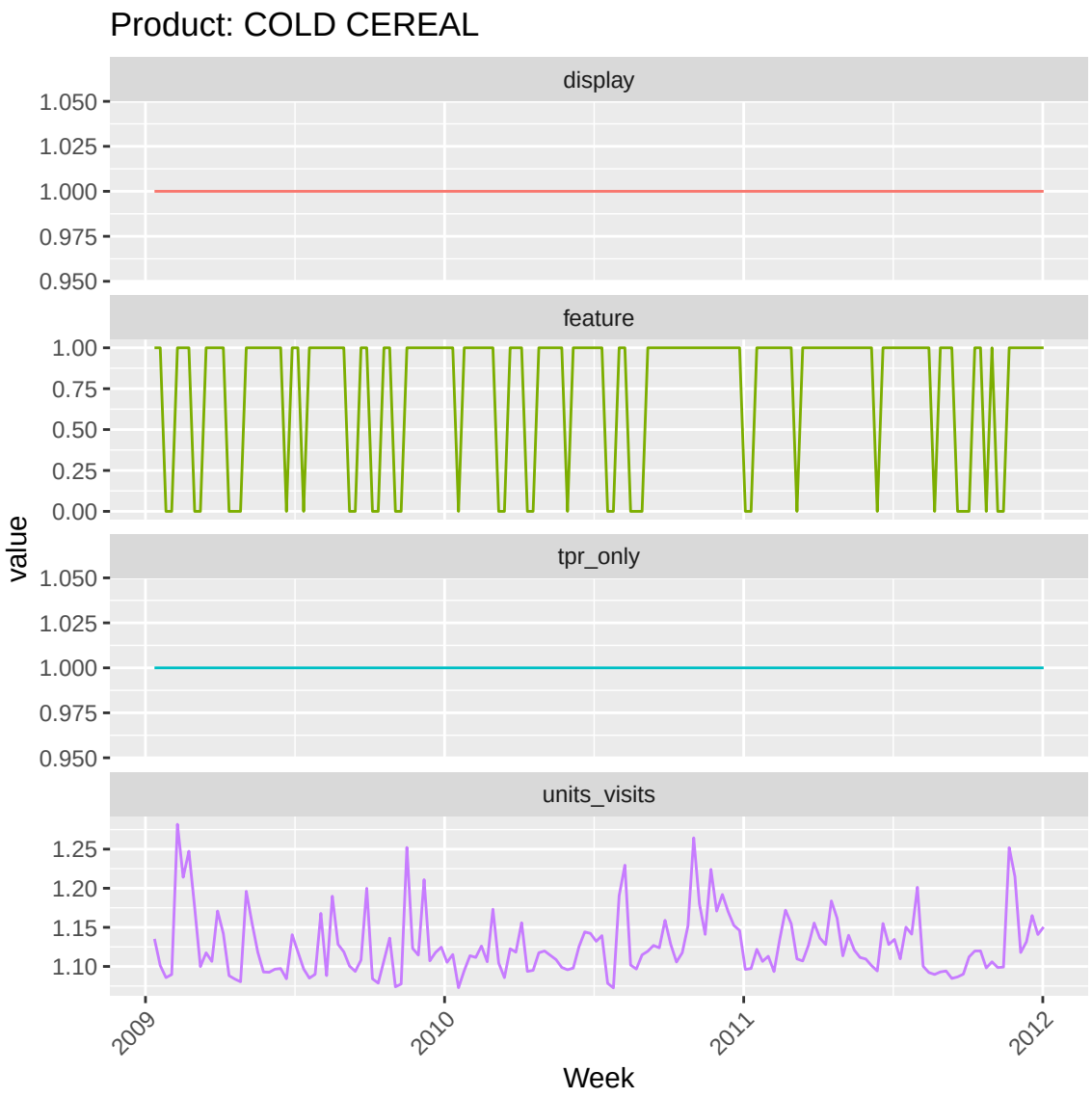
```
units_visits_prom(ts, "BAG SNACKS") |> render_plot()
```



```
units_visits_prom(ts, "ORAL HYGIENE PRODUCTS") |> render_plot()
```



```
units_visits_prom(ts, "COLD CEREAL") |> render_plot()
```



```
units_visits_prom(ts, "FROZEN PIZZA") |> render_plot()
```

In aggregated TS there isn't visible too much correlation among the units/visits and promotions because aggregated product are very noisy in promotional features. Further analysis along each TS individually (disaggregated) is recommended.