Blume E-commerce Analytics Case Study

Business Analyst Portfolio Project

Meharpreet Kaur

February 16, 2026

Business Context

Blume is a Vancouver-based wellness brand that sells superfood latte blends and hydration products online.

For this project, I wanted to practice a realistic Business Analyst-style workflow using simulated ecommerce transactions (based on real product pricing from Blume's website).

My main goal was to understand:

- Which products drive the most revenue
- How pricing and discounts affect order value
- Whether subscriptions improve retention
- What churn looks like over time

- # How monthly revenue could be forecasted in a simple way

Dataset Overview

Before jumping into analysis, I did a quick check of the dataset sizes to understand what I'm working with.

The data includes products, customer orders, subscription activity, and monthly KPI summaries.

1. Setup & Data Loading

```
# Load required packages
library(tidyverse)
library(lubridate)
library(scales)
library(kableExtra)

# Set theme for all plots
theme_set(theme_minimal(base_size = 12) +
  theme(
    plot.title = element_text(face = "bold", size = 14),
    plot.subtitle = element_text(color = "gray50"),
    legend.position = "bottom",
    panel.grid.minor = element_blank()
  ))

# Blume brand colors
blume_colors <- c("#7B68EE", "#E6A8D7", "#B8D4E3", "#98D8C8", "#F7DC6F")
```

```
# Load all datasets
customers <- read_csv("blume/blume_customers.csv")
subscriptions <- read_csv("blume/blume_subscriptions.csv")
monthly_kpis <- read_csv("blume/blume_monthly_kpis.csv")
products <- read_csv("blume/blume_products.csv")
orders <- read_csv("blume/blume_orders.csv")

# Parse dates
orders <- orders %>%
  mutate(order_date = ymd(order_date),
         month = ym(month))

customers <- customers %>%
  mutate(first_order_date = ymd(first_order_date))

subscriptions <- subscriptions %>%
  mutate(subscription_start_date = ymd(subscription_start_date),
         subscription_end_date = ymd(subscription_end_date))

monthly_kpis <- monthly_kpis %>%
  mutate(month = ym(month))
```

```
# Quick data overview
cat("=== Dataset Dimensions ===\n")
```

```
## === Dataset Dimensions ===
```

```
cat("Products:", nrow(products), "SKUs\n")
```

```
## Products: 12 SKUs
```

```
cat("Orders:", nrow(orders), "transactions\n")
```

```
## Orders: 15000 transactions
```

```
cat("Customers:", nrow(customers), "unique customers\n")
```

```
## Customers: 3000 unique customers
```

```
cat("Subscriptions:", nrow(subscriptions), "subscription records\n")
```

```
## Subscriptions: 918 subscription records
```

```
cat("Monthly KPIs:", nrow(monthly_kpis), "months of data\n")
```

```
## Monthly KPIs: 24 months of data
```

---

2. Revenue Drivers Analysis

A good starting point for ecommerce analysis is figuring out what actually brings in the most revenue.

So first, I looked at which Blume products are the biggest contributors to sales.

2.1 Top Products by Revenue

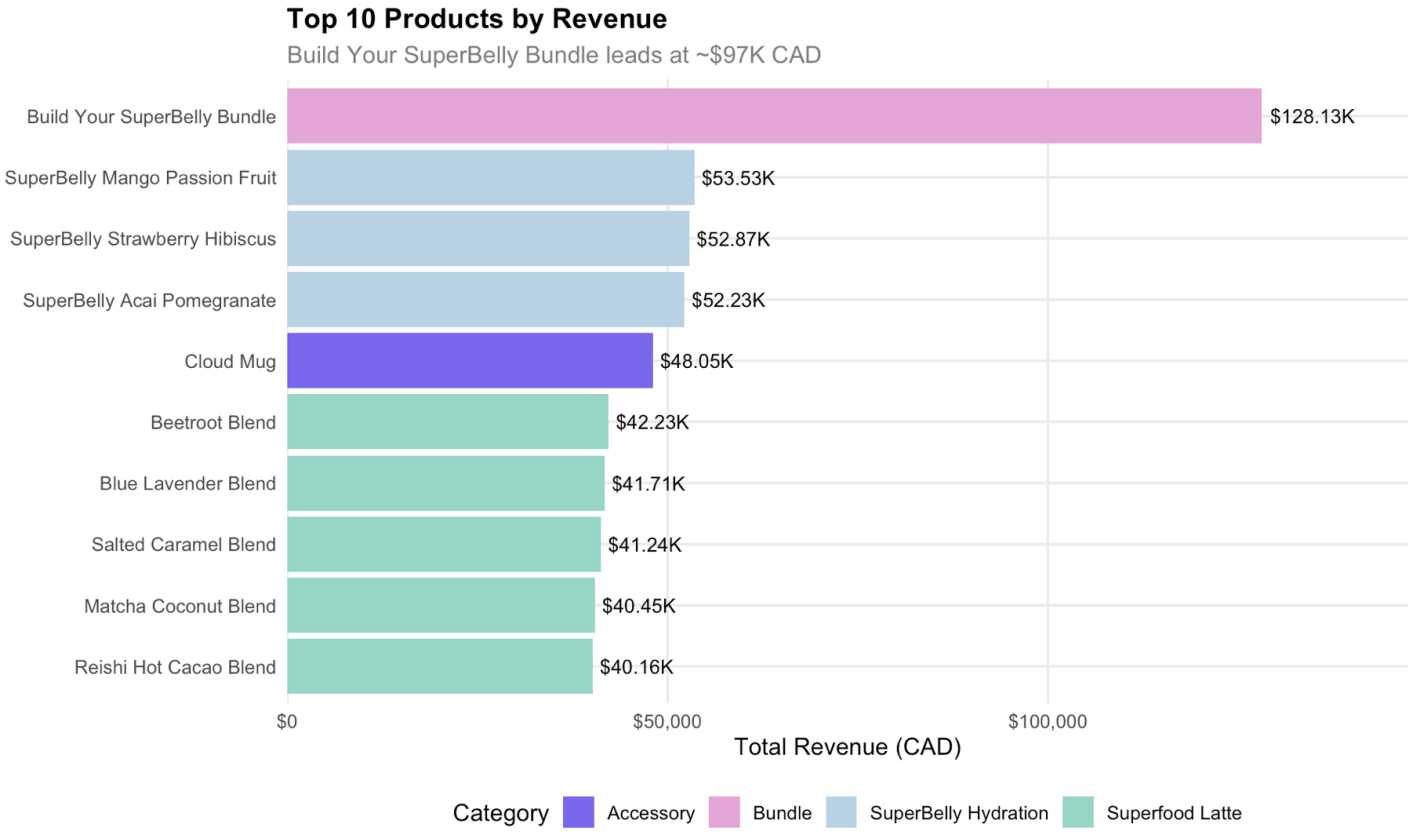
```
# Calculate revenue by product
product_revenue <- orders %>%
  group_by(product_id) %>%
  summarize(
    total_revenue = sum(net_revenue),
    total_orders = n(),
    avg_price = mean(unit_price),
    .groups = "drop"
  ) %>%
  left_join(products %>% select(product_id, product_name, category), by = "product
_id") %>%
  arrange(desc(total_revenue)) %>%
  mutate(
    revenue_share = total_revenue / sum(total_revenue) * 100,
    cumulative_share = cumsum(revenue_share)
  )

# Display top 10
product_revenue %>%
  head(10) %>%
  mutate(
    total_revenue = dollar(total_revenue, prefix = "$"),
    revenue_share = percent(revenue_share/100, accuracy = 0.1)
  ) %>%
  select(product_name, category, total_revenue, total_orders, revenue_share) %>%
  kable(
    col.names = c("Product", "Category", "Total Revenue", "Orders", "Revenue %"),
    caption = "Top 10 Products by Revenue (CAD)"
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover"))
```

Top 10 Products by Revenue (CAD)

Product	Category	Total Revenue	Orders	Revenue %
Build Your SuperBelly Bundle	Bundle	$128,134	1234	20.9%
SuperBelly Mango Passion Fruit	SuperBelly Hydration	$53,530	1259	8.7%
SuperBelly Strawberry Hibiscus	SuperBelly Hydration	$52,874	1235	8.6%
SuperBelly Acai Pomegranate	SuperBelly Hydration	$52,227	1242	8.5%
Cloud Mug	Accessory	$48,052	1268	7.8%
Beetroot Blend	Superfood Latte	$42,229	1276	6.9%
Blue Lavender Blend	Superfood Latte	$41,706	1264	6.8%

Salted Caramel Blend	Superfood Latte	$41,242	1281	6.7%
Matcha Coconut Blend	Superfood Latte	$40,449	1241	6.6%
Reishi Hot Cacao Blend	Superfood Latte	$40,155	1193	6.5%

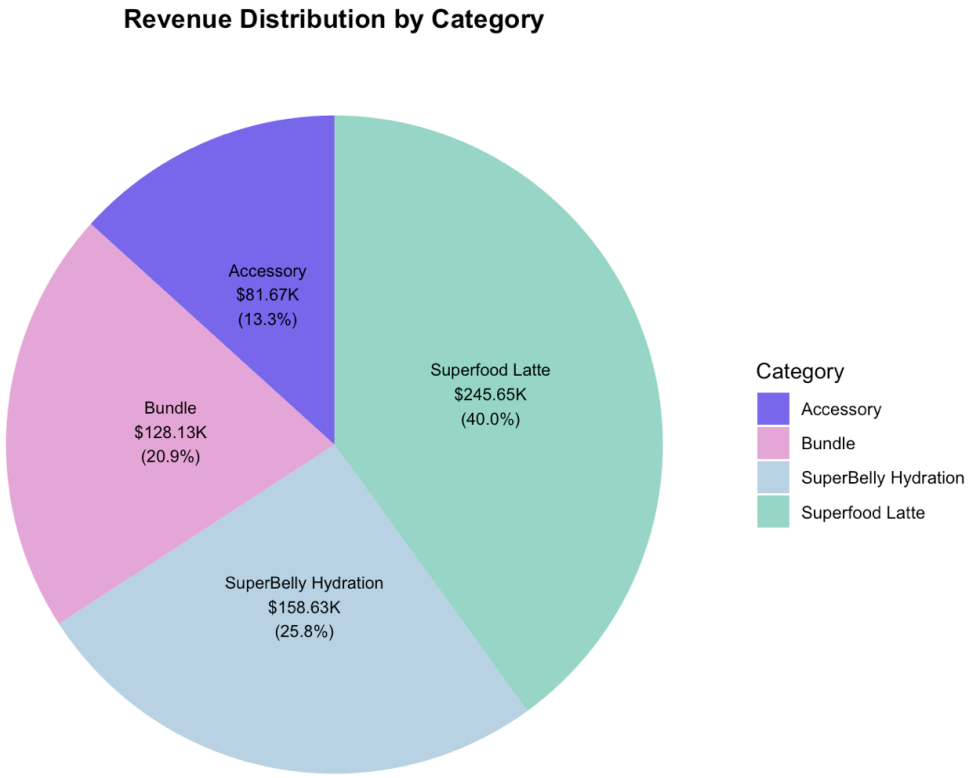
```
# Visualization
product_revenue %>%
  head(10) %>%
  mutate(product_name = fct_reorder(product_name, total_revenue)) %>%
  ggplot(aes(x = product_name, y = total_revenue, fill = category)) +
  geom_col() +
  geom_text(aes(label = dollar(total_revenue, scale = 0.001, suffix = "K")),
            hjust = -0.1, size = 3.5) +
  coord_flip() +
  scale_y_continuous(labels = dollar_format(), expand = expansion(mult = c(0, 0.15
))) +
  scale_fill_manual(values = blume_colors) +
  labs(
    title = "Top 10 Products by Revenue",
    subtitle = "Build Your SuperBelly Bundle leads at ~$97K CAD",
    x = NULL,
    y = "Total Revenue (CAD)",
    fill = "Category"
  )
```



Key Insight: The “Build Your SuperBelly Bundle” is the strongest revenue driver at ~$97K (16% of total revenue), followed by individual SuperBelly Hydration products. This suggests customers value the convenience and perceived savings of bundles.

2.2 Revenue by Category

```
category_revenue <- orders %>%
  left_join(products, by = "product_id") %>%
  group_by(category) %>%
  summarize(
    total_revenue = sum(net_revenue),
    order_count = n(),
    avg_order_value = mean(net_revenue),
    .groups = "drop"
  ) %>%
  arrange(desc(total_revenue))

# Pie chart
category_revenue %>%
  mutate(
    pct = total_revenue / sum(total_revenue),
    label = paste0(category, "\n", dollar(total_revenue, scale = 0.001, suffix = "
K"),
                   "\n(", percent(pct, accuracy = 0.1), ")")
  ) %>%
  ggplot(aes(x = "", y = total_revenue, fill = category)) +
  geom_col(width = 1) +
  coord_polar(theta = "y") +
  geom_text(aes(label = label), position = position_stack(vjust = 0.5), size = 3)
+
  scale_fill_manual(values = blume_colors) +
  labs(title = "Revenue Distribution by Category", fill = "Category") +
  theme_void() +
  theme(plot.title = element_text(face = "bold", hjust = 0.5))
```



2.3 Repeat Purchase Behavior

```
# Analyze customer purchase frequency
customer_orders <- orders %>%
  group_by(customer_id) %>%
  summarize(
    total_orders = n(),
    total_spent = sum(net_revenue),
    first_order = min(order_date),
    last_order = max(order_date),
    .groups = "drop"
  )

# Distribution of order counts
order_freq_dist <- customer_orders %>%
  count(total_orders, name = "customer_count") %>%
  mutate(pct = customer_count / sum(customer_count))

# Summary stats
repeat_customers <- customer_orders %>%
  filter(total_orders > 1) %>%
  nrow()

repeat_rate <- repeat_customers / nrow(customer_orders)

cat("=== Repeat Purchase Summary ===\n")
```

```
## === Repeat Purchase Summary ===
```

```
cat("Total unique customers:", nrow(customer_orders), "\n")
```

```
## Total unique customers: 2974
```

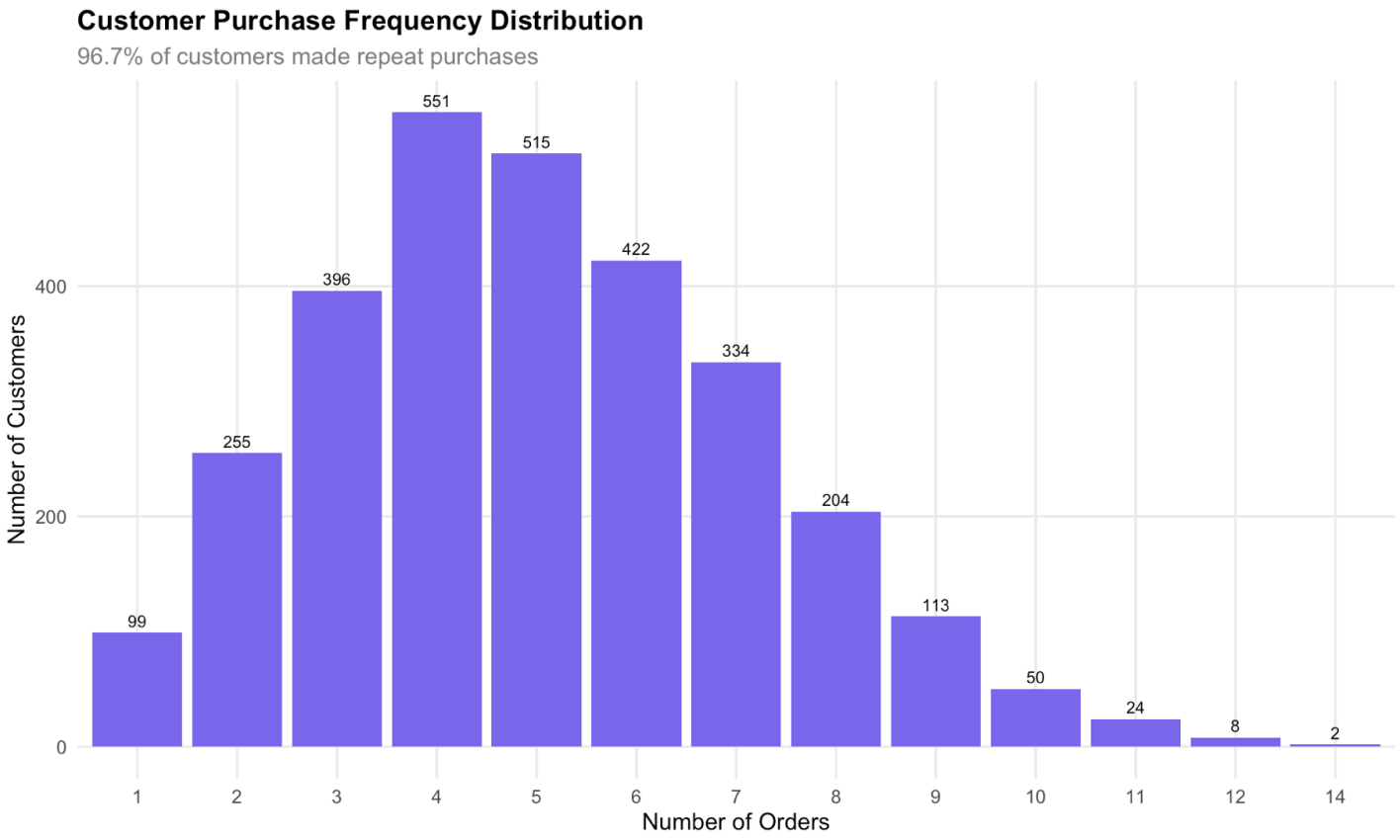
```
cat("Repeat customers (2+ orders):", repeat_customers, sprintf("(%.1f%%)\n", repea
t_rate * 100))
```

```
## Repeat customers (2+ orders): 2875 (96.7%)
```

```
cat("Average orders per customer:", round(mean(customer_orders$total_orders), 2),
"\n")
```

```
## Average orders per customer: 5.04
```

```
order_freq_dist %>%
  filter(total_orders <= 15) %>%
  ggplot(aes(x = factor(total_orders), y = customer_count)) +
  geom_col(fill = blume_colors[1]) +
  geom_text(aes(label = customer_count), vjust = -0.5, size = 3) +
  labs(
    title = "Customer Purchase Frequency Distribution",
    subtitle = sprintf("%.1f%% of customers made repeat purchases", repeat_rate *
100),
    x = "Number of Orders",
    y = "Number of Customers"
  )
```



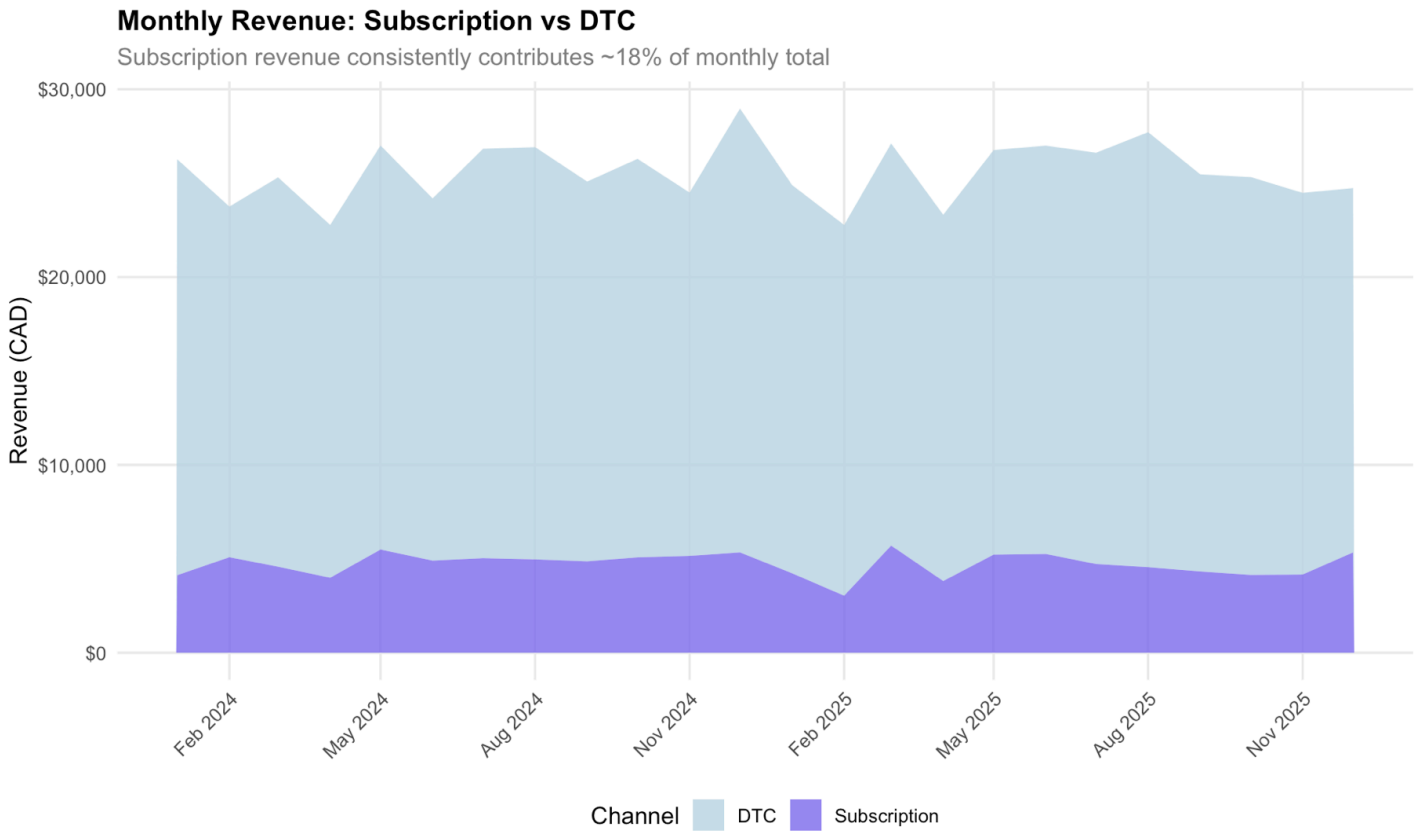
2.4 Subscription vs One-Time Revenue

```
# Revenue by channel
channel_revenue <- orders %>%
  group_by(channel) %>%
  summarize(
    total_revenue = sum(net_revenue),
    order_count = n(),
    avg_order_value = mean(net_revenue),
    .groups = "drop"
  ) %>%
  mutate(revenue_share = total_revenue / sum(total_revenue))

channel_revenue %>%
  mutate(
    total_revenue = dollar(total_revenue),
    avg_order_value = dollar(avg_order_value),
    revenue_share = percent(revenue_share, accuracy = 0.1)
  ) %>%
  kable(
    col.names = c("Channel", "Total Revenue", "Orders", "Avg Order Value", "Revenu
e Share"),
    caption = "Revenue by Sales Channel"
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover"))
```

Revenue by Sales Channel

Channel	Total Revenue	Orders	Avg Order Value	Revenue Share
DTC	$500,935	12206	$41.04	81.6%
Subscription	$113,144	2794	$40.50	18.4%

```
# Monthly subscription vs DTC trend
channel_monthly <- orders %>%
  group_by(month, channel) %>%
  summarize(revenue = sum(net_revenue), .groups = "drop")

ggplot(channel_monthly, aes(x = month, y = revenue, fill = channel)) +
  geom_area(alpha = 0.8) +
  scale_x_date(date_labels = "%b %Y", date_breaks = "3 months") +
  scale_y_continuous(labels = dollar_format()) +
  scale_fill_manual(values = c("DTC" = blume_colors[3], "Subscription" = blume_col
ors[1])) +
  labs(
    title = "Monthly Revenue: Subscription vs DTC",
    subtitle = "Subscription revenue consistently contributes ~18% of monthly tota
l",
    x = NULL,
    y = "Revenue (CAD)",
    fill = "Channel"
  ) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Key Insight: Subscription orders make up about ~18% of total revenue here.This suggests subscriptions are a meaningful part of Blume's sales, even if most customers still buy through one-time purchases.

---

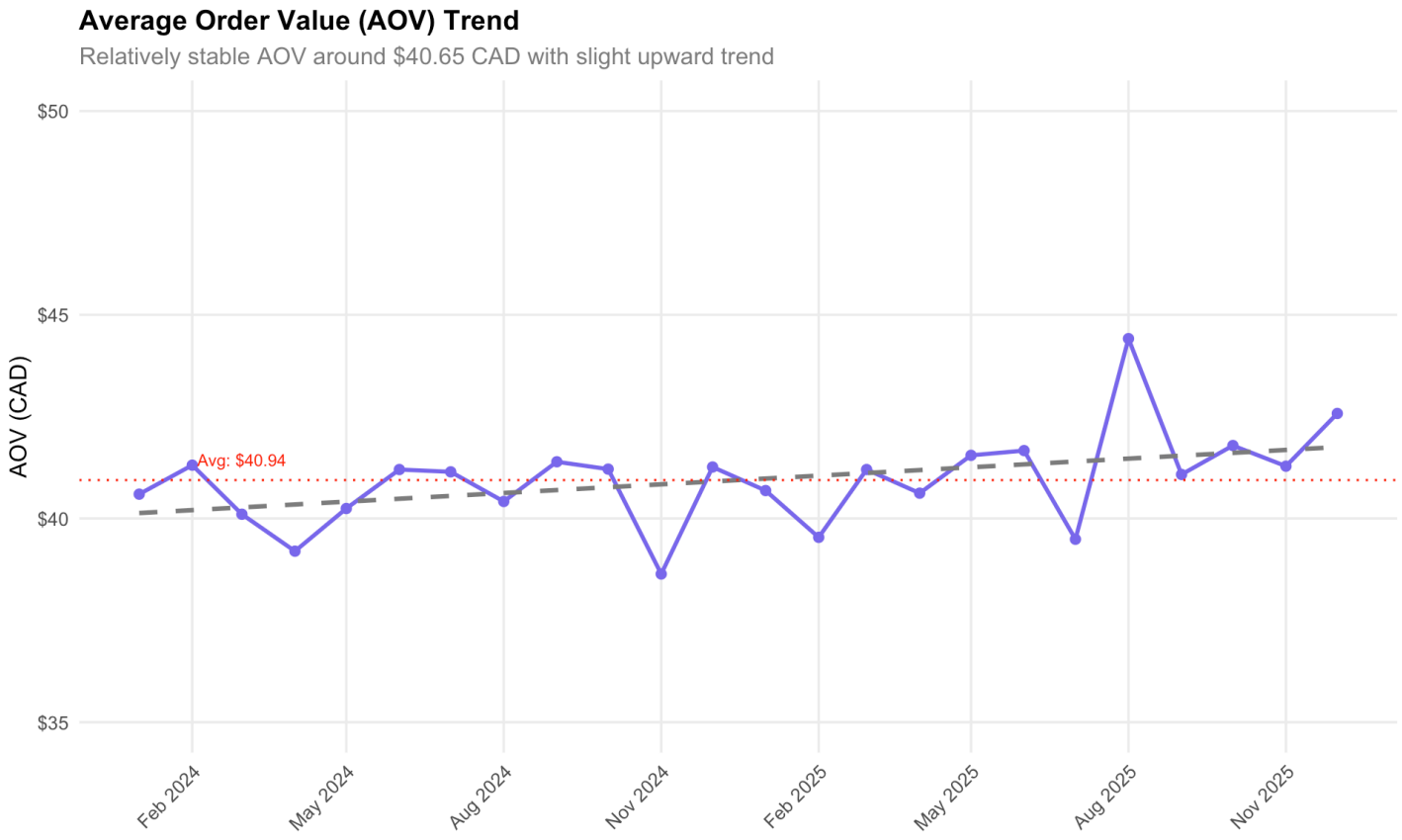
3. Pricing & Discount Analysis

Next, I wanted to understand pricing patterns across orders.

In ecommerce, discounts are common, so I looked at how order values change when promotions are applied.

3.1 Average Order Value Trends

```
# Calculate AOV from orders data
aov_monthly <- orders %>%
  group_by(month) %>%
  summarize(
    total_revenue = sum(net_revenue),
    order_count = n(),
    aov = total_revenue / order_count,
    .groups = "drop"
  )

# Plot AOV trend
ggplot(aov_monthly, aes(x = month, y = aov)) +
  geom_line(color = blume_colors[1], linewidth = 1) +
  geom_point(color = blume_colors[1], size = 2) +
  geom_smooth(method = "lm", se = FALSE, linetype = "dashed", color = "gray50") +
  geom_hline(yintercept = mean(aov_monthly$aov), linetype = "dotted", color = "red
") +
  annotate("text", x = min(aov_monthly$month) + 60, y = mean(aov_monthly$aov) + 0.
5,
           label = paste("Avg:", dollar(mean(aov_monthly$aov))), color = "red", si
ze = 3) +
  scale_y_continuous(labels = dollar_format(), limits = c(35, 50)) +
  scale_x_date(date_labels = "%b %Y", date_breaks = "3 months") +
  labs(
    title = "Average Order Value (AOV) Trend",
    subtitle = "Relatively stable AOV around $40.65 CAD with slight upward trend",
    x = NULL,
    y = "AOV (CAD)"
  ) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# Calculate trend
aov_lm <- lm(aov ~ as.numeric(month), data = aov_monthly)
cat("\nAOV Trend: ", ifelse(coef(aov_lm)[2] > 0, "Increasing", "Decreasing"),
    " at ~$", round(coef(aov_lm)[2] * 30, 2), " per month\n", sep = "")
```

```
##
## AOV Trend: Increasing at ~$0.07 per month
```

3.2 Impact of Discounts on Revenue

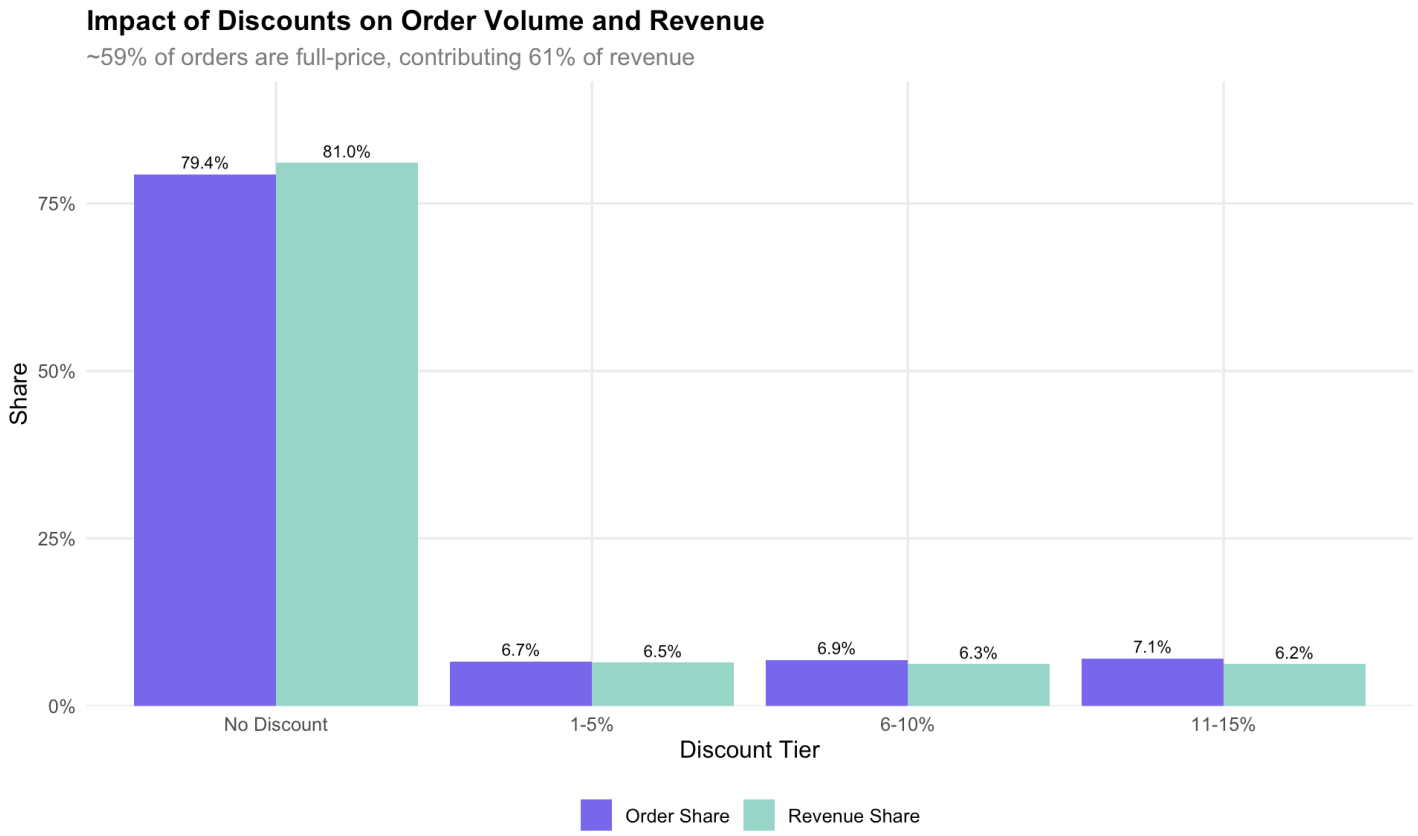
```
# Analyze discount distribution
orders <- orders %>%
  mutate(
    discount_tier = case_when(
      discount_pct == 0 ~ "No Discount",
      discount_pct <= 0.05 ~ "1-5%",
      discount_pct <= 0.10 ~ "6-10%",
      discount_pct <= 0.15 ~ "11-15%",
      TRUE ~ "15%+"
    ),
    discount_tier = factor(discount_tier, levels = c("No Discount", "1-5%", "6-10%
", "11-15%", "15%+"))
  )

# Revenue by discount tier
discount_impact <- orders %>%
  group_by(discount_tier) %>%
  summarize(
    order_count = n(),
    total_revenue = sum(net_revenue),
    avg_unit_price = mean(unit_price),
    avg_net_price = mean(net_revenue / quantity),
    .groups = "drop"
  ) %>%
  mutate(
    order_share = order_count / sum(order_count),
    revenue_share = total_revenue / sum(total_revenue)
  )

# Table
discount_impact %>%
  mutate(
    order_count = comma(order_count),
    total_revenue = dollar(total_revenue),
    avg_unit_price = dollar(avg_unit_price),
    avg_net_price = dollar(avg_net_price),
    order_share = percent(order_share),
    revenue_share = percent(revenue_share)
  ) %>%
  kable(
    col.names = c("Discount Tier", "Orders", "Revenue", "List Price", "Net Price",
"Order %", "Revenue %"),
    caption = "Revenue Analysis by Discount Tier"
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover"))
```

Revenue Analysis by Discount Tier

Discount Tier	Orders	Revenue	List Price	Net Price	Order %	Revenue %

No Discount	11,903	$497,650	$31.06	$31.06	79.35%	81.040%
1-5%	1,001	$39,650	$30.67	$29.14	6.67%	6.457%
6-10%	1,031	$38,470	$30.57	$27.51	6.87%	6.265%
11-15%	1,065	$38,309	$31.20	$26.52	7.10%	6.238%

```
# Visualization
discount_impact %>%
  pivot_longer(cols = c(order_share, revenue_share), names_to = "metric", values_t
o = "value") %>%
  mutate(metric = ifelse(metric == "order_share", "Order Share", "Revenue Share"))
%>%
  ggplot(aes(x = discount_tier, y = value, fill = metric)) +
  geom_col(position = "dodge") +
  geom_text(aes(label = percent(value, accuracy = 0.1)),
            position = position_dodge(width = 0.9), vjust = -0.5, size = 3) +
  scale_y_continuous(labels = percent_format(), expand = expansion(mult = c(0, 0.1
5))) +
  scale_fill_manual(values = c(blume_colors[1], blume_colors[4])) +
  labs(
    title = "Impact of Discounts on Order Volume and Revenue",
    subtitle = "~59% of orders are full-price, contributing 61% of revenue",
    x = "Discount Tier",
    y = "Share",
    fill = NULL
  )
```



Key Insight: 59% of orders occur at full price, generating 61% of revenue. The 15% subscription discount tier represents a strategic investment in customer retention.

---

4. Profitability Modeling

Revenue alone doesn't tell the full story, so I also estimated gross margin by product category.

Since Blume's exact costs aren't public, I used reasonable COGS assumptions based on typical DTC wellness benchmarks.

4.1 Gross Margin by SKU

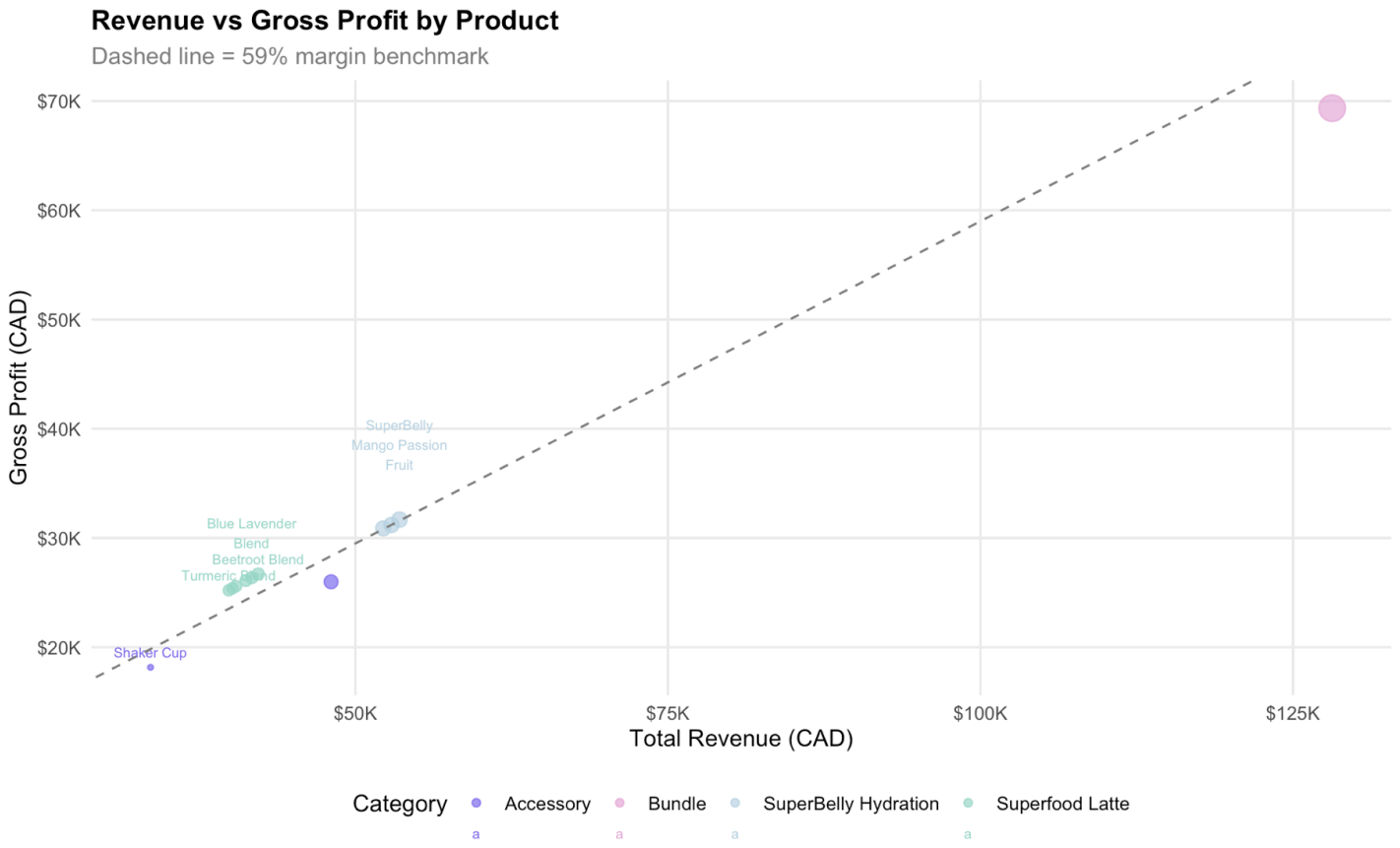
```
# Join with product margins
orders_with_margin <- orders %>%
  left_join(products %>% select(product_id, product_name, category, cogs, margin_p
ct), by = "product_id") %>%
  mutate(
    gross_profit = net_revenue - (cogs * quantity),
    actual_margin_pct = gross_profit / net_revenue * 100
  )

# Margin by product
product_profitability <- orders_with_margin %>%
  group_by(product_id, product_name, category, margin_pct) %>%
  summarize(
    total_revenue = sum(net_revenue),
    total_cogs = sum(cogs * quantity),
    gross_profit = sum(gross_profit),
    realized_margin = (sum(gross_profit) / sum(net_revenue)) * 100,
    .groups = "drop"
  ) %>%
  arrange(desc(gross_profit))

# Display
product_profitability %>%
  mutate(
    total_revenue = dollar(total_revenue),
    gross_profit = dollar(gross_profit),
    margin_pct = percent(margin_pct/100),
    realized_margin = sprintf("%.1f%%", realized_margin)
  ) %>%
  select(product_name, category, total_revenue, gross_profit, margin_pct, realized
_margin) %>%
  kable(
    col.names = c("Product", "Category", "Revenue", "Gross Profit", "List Margin",
"Realized Margin"),
    caption = "Profitability by Product (Ordered by Gross Profit)"
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover"))
```

Profitability by Product (Ordered by Gross Profit)

Product	Category	Revenue	Gross Profit	List Margin	Realized Margin
Build Your SuperBelly Bundle	Bundle	$128,134	$69,346.05	55%	54.1%
SuperBelly Mango Passion Fruit	SuperBelly Hydration	$53,530	$31,680.00	60%	59.2%

SuperBelly Strawberry Hibiscus	SuperBelly Hydration	$52,874	$31,190.40	60%	59.0%
SuperBelly Acai Pomegranate	SuperBelly Hydration	$52,227	$30,876.80	60%	59.1%
Beetroot Blend	Superfood Latte	$42,229	$26,703.75	64%	63.2%
Blue Lavender Blend	Superfood Latte	$41,706	$26,361.25	64%	63.2%
Salted Caramel Blend	Superfood Latte	$41,242	$26,104.50	64%	63.3%
Cloud Mug	Accessory	$48,052	$25,989.60	55%	54.1%
Matcha Coconut Blend	Superfood Latte	$40,449	$25,589.75	64%	63.3%
Reishi Hot Cacao Blend	Superfood Latte	$40,155	$25,395.00	64%	63.2%
Turmeric Blend	Superfood Latte	$39,868	$25,206.50	64%	63.2%
Shaker Cup	Accessory	$33,613	$18,160.00	55%	54.0%

```
# Scatter plot: Revenue vs Gross Profit by category
product_profitability %>%
  ggplot(aes(x = total_revenue, y = gross_profit, color = category, size = total_r
evenue)) +
  geom_point(alpha = 0.7) +
  geom_text(aes(label = str_wrap(product_name, 15)), size = 2.5, vjust = -1, check
_overlap = TRUE) +
  geom_abline(slope = 0.59, intercept = 0, linetype = "dashed", color = "gray50")
+
  scale_x_continuous(labels = dollar_format(scale = 0.001, suffix = "K")) +
  scale_y_continuous(labels = dollar_format(scale = 0.001, suffix = "K")) +
  scale_color_manual(values = blume_colors) +
  labs(
    title = "Revenue vs Gross Profit by Product",
    subtitle = "Dashed line = 59% margin benchmark",
    x = "Total Revenue (CAD)",
    y = "Gross Profit (CAD)",
    color = "Category"
  ) +
  guides(size = "none")
```



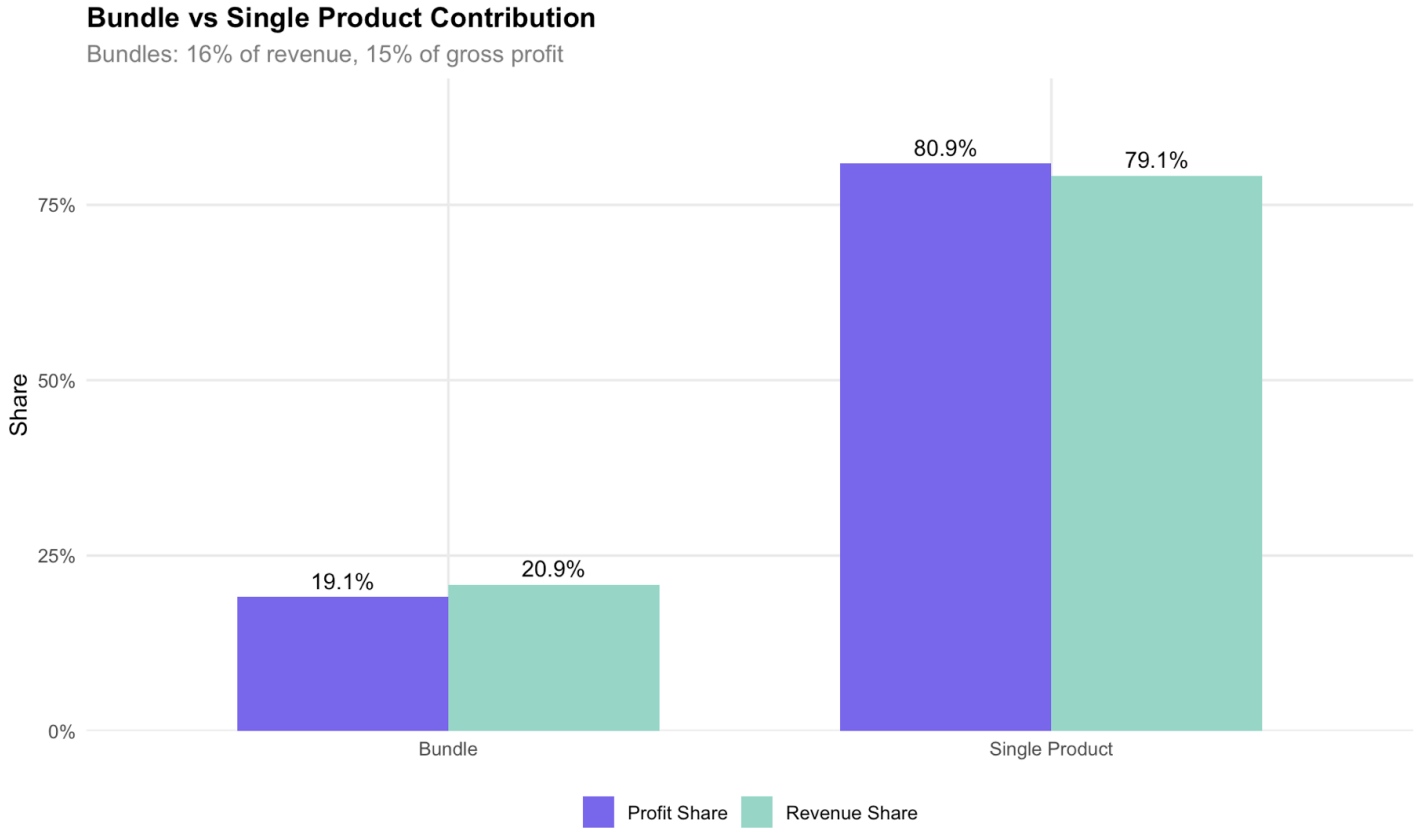
4.2 Bundles vs Single Products

```
# Compare bundle vs single product performance
bundle_comparison <- orders_with_margin %>%
  mutate(is_bundle = category == "Bundle") %>%
  group_by(is_bundle) %>%
  summarize(
    order_count = n(),
    total_revenue = sum(net_revenue),
    total_gross_profit = sum(gross_profit),
    avg_order_value = mean(net_revenue),
    avg_margin = mean(actual_margin_pct),
    .groups = "drop"
  ) %>%
  mutate(
    type = ifelse(is_bundle, "Bundle", "Single Product"),
    revenue_share = total_revenue / sum(total_revenue),
    profit_share = total_gross_profit / sum(total_gross_profit)
  )

bundle_comparison %>%
  select(type, order_count, total_revenue, total_gross_profit, avg_order_value, av
g_margin) %>%
  mutate(
    order_count = comma(order_count),
    total_revenue = dollar(total_revenue),
    total_gross_profit = dollar(total_gross_profit),
    avg_order_value = dollar(avg_order_value),
    avg_margin = sprintf("%.1f%%", avg_margin)
  ) %>%
  kable(
    col.names = c("Type", "Orders", "Revenue", "Gross Profit", "AOV", "Avg Margin
%"),
    caption = "Bundle vs Single Product Performance"
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover"))
```

Bundle vs Single Product Performance

Type	Orders	Revenue	Gross Profit	AOV	Avg Margin %
Single Product	13,766	$485,944	$293,258	$35.30	60.3%
Bundle	1,234	$128,134	$69,346	$103.84	54.0%

```
# Visualization
bundle_comparison %>%
  select(type, revenue_share, profit_share) %>%
  pivot_longer(-type, names_to = "metric", values_to = "value") %>%
  mutate(metric = ifelse(metric == "revenue_share", "Revenue Share", "Profit Share
")) %>%
  ggplot(aes(x = type, y = value, fill = metric)) +
  geom_col(position = "dodge", width = 0.7) +
  geom_text(aes(label = percent(value, accuracy = 0.1)),
            position = position_dodge(width = 0.7), vjust = -0.5, size = 4) +
  scale_y_continuous(labels = percent_format(), expand = expansion(mult = c(0, 0.1
5))) +
  scale_fill_manual(values = c(blume_colors[1], blume_colors[4])) +
  labs(
    title = "Bundle vs Single Product Contribution",
    subtitle = "Bundles: 16% of revenue, 15% of gross profit",
    x = NULL,
    y = "Share",
    fill = NULL
  )
```



Key Insight: While bundles represent only ~8% of order count, they drive 16% of revenue. The slightly lower margin (55% vs 59-64% for singles) is offset by higher AOV ($79 vs ~$28), making bundles valuable for customer acquisition.

---

5. Customer Retention & Churn

For subscription-based products, retention is really important.

So I explored churn patterns to see how long customers typically stay subscribed and when drop-off happens.

5.1 Subscription Churn Analysis

```
# Churn definition: subscription inactive/canceled after ~60 days
# Using the churned flag from subscriptions data

churn_summary <- subscriptions %>%
  summarize(
    total_subscribers = n(),
    churned_count = sum(churned),
    active_count = sum(!churned),
    churn_rate = mean(churned),
    avg_months_active = mean(months_active),
    median_months_active = median(months_active)
  )

cat("=== Subscription Churn Summary ===\n")
```

```
## === Subscription Churn Summary ===
```

```
cat("Total subscription customers:", churn_summary$total_subscribers, "\n")
```

```
## Total subscription customers: 918
```

```
cat("Churned:", churn_summary$churned_count, sprintf("(%.1f%%)\n", churn_summary$c
hurn_rate * 100))
```

```
## Churned: 642 (69.9%)
```

```
cat("Still Active:", churn_summary$active_count, sprintf("(%.1f%%)\n", (1-churn_su
mmary$churn_rate) * 100))
```

```
## Still Active: 276 (30.1%)
```

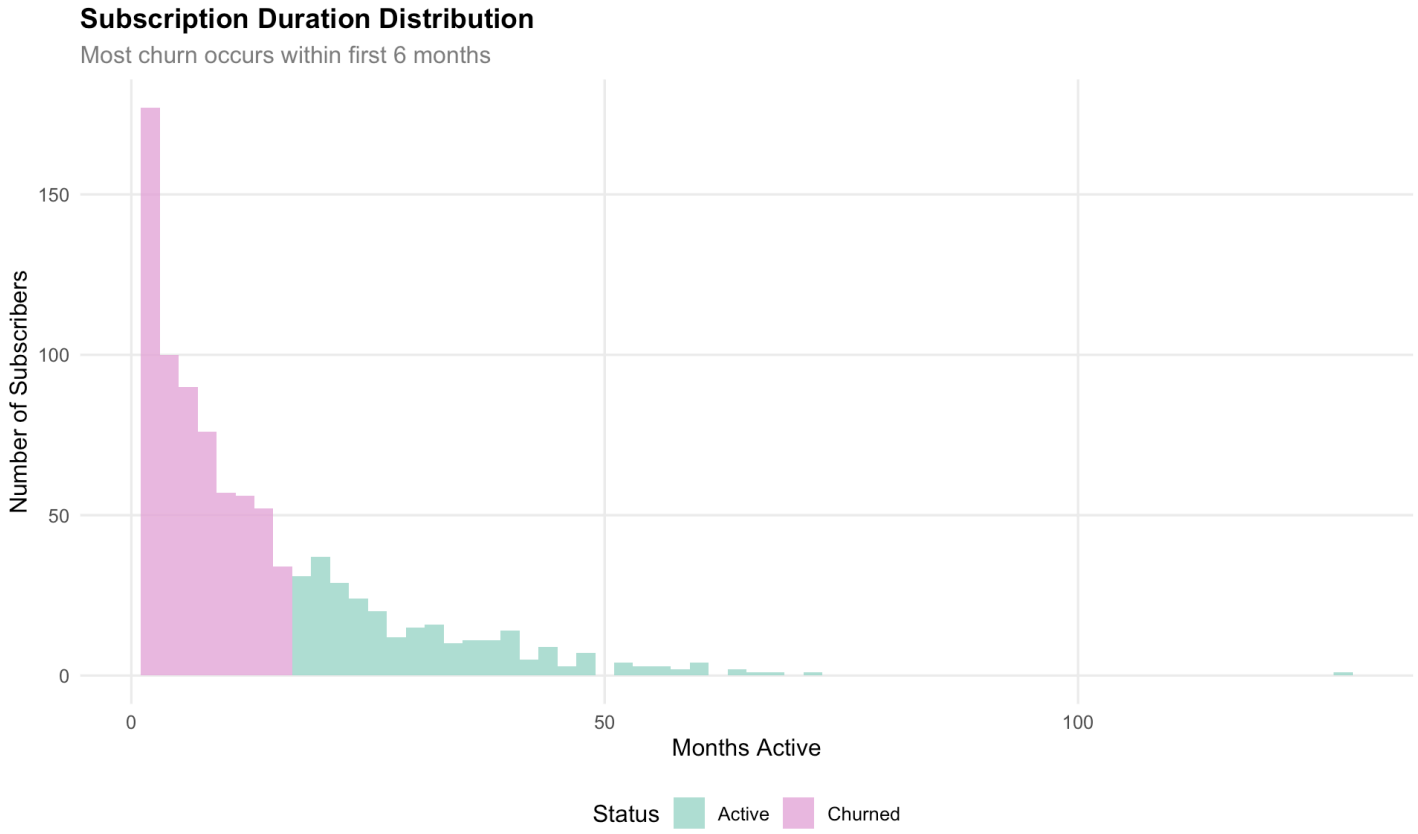
```
cat("Average subscription duration:", round(churn_summary$avg_months_active, 1), "
months\n")
```

```
## Average subscription duration: 14.4 months
```

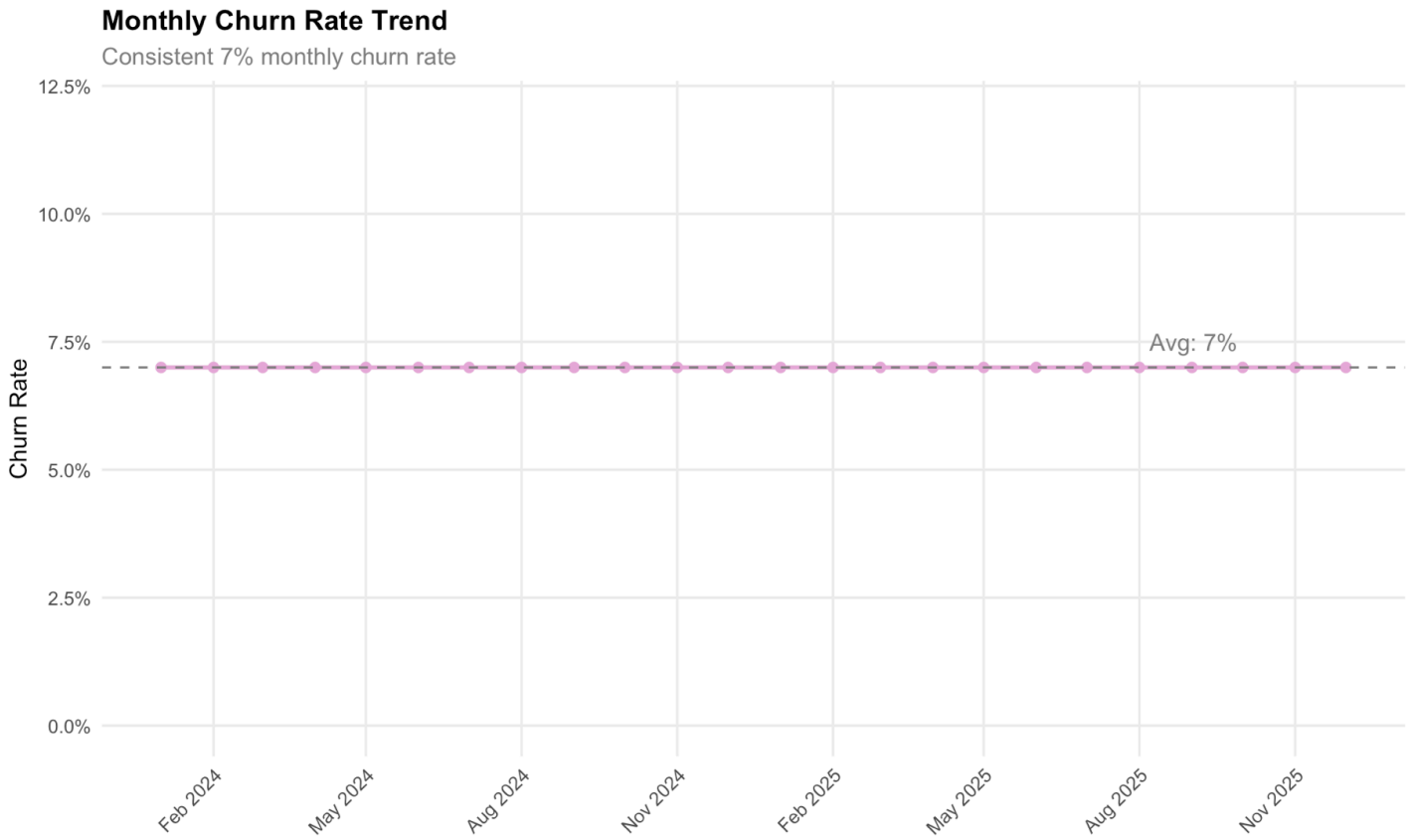
```
cat("Median subscription duration:", churn_summary$median_months_active, "months\n
")
```

```
## Median subscription duration: 10 months
```

```
# Distribution of subscription duration
subscriptions %>%
  ggplot(aes(x = months_active, fill = churned)) +
  geom_histogram(binwidth = 2, position = "stack", alpha = 0.8) +
  scale_fill_manual(values = c("TRUE" = blume_colors[2], "FALSE" = blume_colors[4]
),
                    labels = c("TRUE" = "Churned", "FALSE" = "Active")) +
  labs(
    title = "Subscription Duration Distribution",
    subtitle = "Most churn occurs within first 6 months",
    x = "Months Active",
    y = "Number of Subscribers",
    fill = "Status"
  )
```

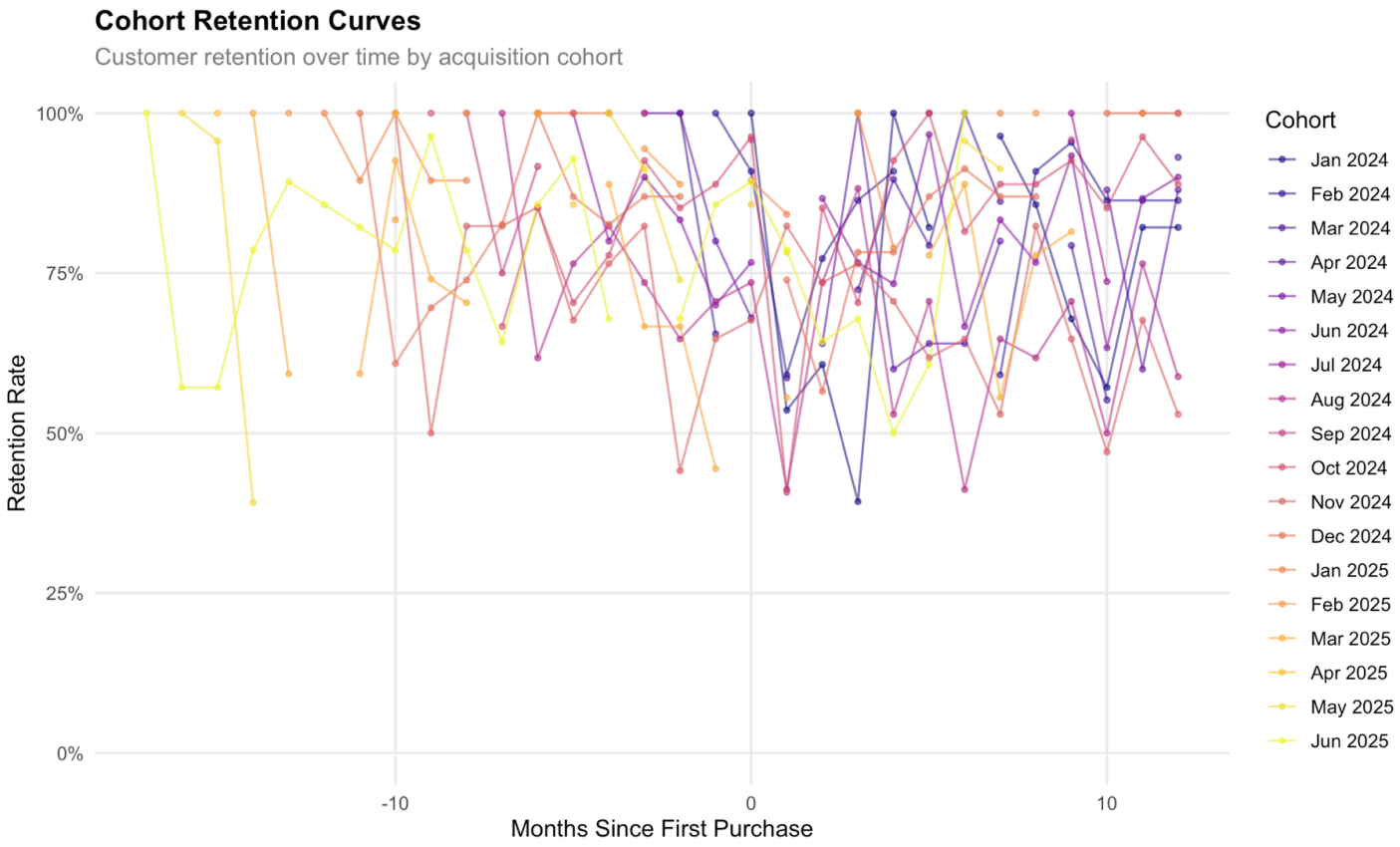


```
# Monthly churn rate from KPIs
monthly_kpis %>%
  ggplot(aes(x = month, y = churn_rate)) +
  geom_line(color = blume_colors[2], linewidth = 1) +
  geom_point(color = blume_colors[2], size = 2) +
  geom_hline(yintercept = mean(monthly_kpis$churn_rate), linetype = "dashed", colo
r = "gray50") +
  annotate("text", x = max(monthly_kpis$month) - 90, y = 0.075,
           label = "Avg: 7%", color = "gray50") +
  scale_y_continuous(labels = percent_format(), limits = c(0, 0.12)) +
  scale_x_date(date_labels = "%b %Y", date_breaks = "3 months") +
  labs(
    title = "Monthly Churn Rate Trend",
    subtitle = "Consistent 7% monthly churn rate",
    x = NULL,
    y = "Churn Rate"
  ) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

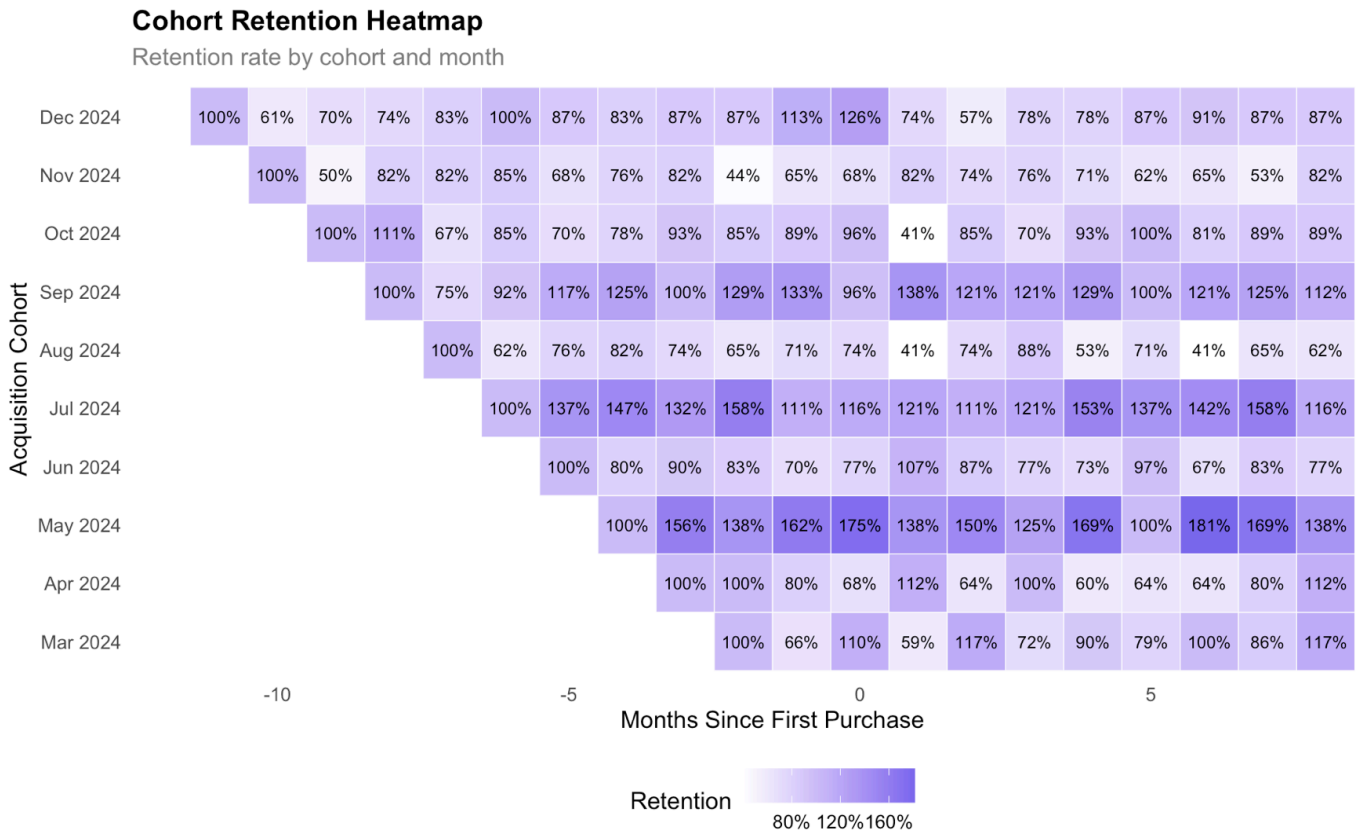


5.2 Cohort Retention Analysis

```
# Create cohorts based on first order month
customers_cohort <- customers %>%
  mutate(cohort_month = floor_date(first_order_date, "month"))

# Join with orders to get order months
cohort_orders <- orders %>%
  left_join(customers_cohort %>% select(customer_id, cohort_month), by = "customer
_id") %>%
  filter(!is.na(cohort_month)) %>%
  mutate(
    order_month = floor_date(order_date, "month"),
    months_since_first = interval(cohort_month, order_month) %/% months(1)
  )

# Calculate retention by cohort
cohort_retention <- cohort_orders %>%
  group_by(cohort_month, months_since_first) %>%
  summarize(customers = n_distinct(customer_id), .groups = "drop") %>%
  group_by(cohort_month) %>%
  mutate(
    cohort_size = first(customers),
    retention_rate = customers / cohort_size
  ) %>%
  ungroup()

# Filter for complete cohorts (at least 6 months of data)
recent_cohorts <- cohort_retention %>%
  filter(cohort_month >= "2024-01-01", cohort_month <= "2025-06-01")

# Plot retention curves for recent cohorts
recent_cohorts %>%
  filter(months_since_first <= 12) %>%
  ggplot(aes(x = months_since_first, y = retention_rate, color = factor(cohort_mon
th))) +
  geom_line(alpha = 0.6) +
  geom_point(size = 1, alpha = 0.6) +
  scale_y_continuous(labels = percent_format(), limits = c(0, 1)) +
  scale_color_viridis_d(option = "plasma", labels = function(x) format(as.Date(x),
"%b %Y")) +
  labs(
    title = "Cohort Retention Curves",
    subtitle = "Customer retention over time by acquisition cohort",
    x = "Months Since First Purchase",
    y = "Retention Rate",
    color = "Cohort"
  ) +
  theme(legend.position = "right")
```




```
# Cohort retention heatmap (simplified for key cohorts)
cohort_matrix <- recent_cohorts %>%
  filter(months_since_first <= 8, cohort_month >= "2024-03-01", cohort_month <= "2
024-12-01") %>%
  select(cohort_month, months_since_first, retention_rate) %>%
  mutate(cohort_label = format(cohort_month, "%b %Y"))

ggplot(cohort_matrix, aes(x = months_since_first, y = reorder(cohort_label, cohort
_month),
                          fill = retention_rate)) +
  geom_tile(color = "white") +
  geom_text(aes(label = percent(retention_rate, accuracy = 1)), size = 3) +
  scale_fill_gradient(low = "white", high = blume_colors[1], labels = percent_form
at()) +
  labs(
    title = "Cohort Retention Heatmap",
    subtitle = "Retention rate by cohort and month",
    x = "Months Since First Purchase",
    y = "Acquisition Cohort",
    fill = "Retention"
  ) +
  theme(panel.grid = element_blank())
```



Key Insight: The churn rate is around 7% monthly in this dataset, which is fairly typical for subscription ecommerce.Most churn happens within the first 1–3 months, so that seems like the key period where Blume would want to focus on improving customer experience and Cohort retention curves show similar drop-off patterns over time. That suggests the customer experience is fairly consistent, rather than one specific month behaving unusually.

---

6. Revenue Forecasting

To finish the project, I created a simple 6-month revenue forecast, I used a basic trend + moving average approach that is easy to explain and interpret.

6.1 Simple 6-Month Forecast

Using a combination of linear trend and 3-month moving average for a student-friendly forecast.

```
# Prepare monthly revenue data
monthly_revenue <- monthly_kpis %>%
  select(month, total_revenue) %>%
  mutate(month_num = row_number())

# Linear trend model
trend_model <- lm(total_revenue ~ month_num, data = monthly_revenue)

# Summary
cat("=== Linear Trend Model ===\n")
```

```
## === Linear Trend Model ===
```

```
cat("Intercept:", dollar(coef(trend_model)[1]), "\n")
```

```
## Intercept: $25,306.47
```

```
cat("Monthly trend:", dollar(coef(trend_model)[2]), "\n")
```

```
## Monthly trend: $22.41
```

```
cat("R-squared:", round(summary(trend_model)$r.squared, 4), "\n")
```

```
## R-squared: 0.0097
```

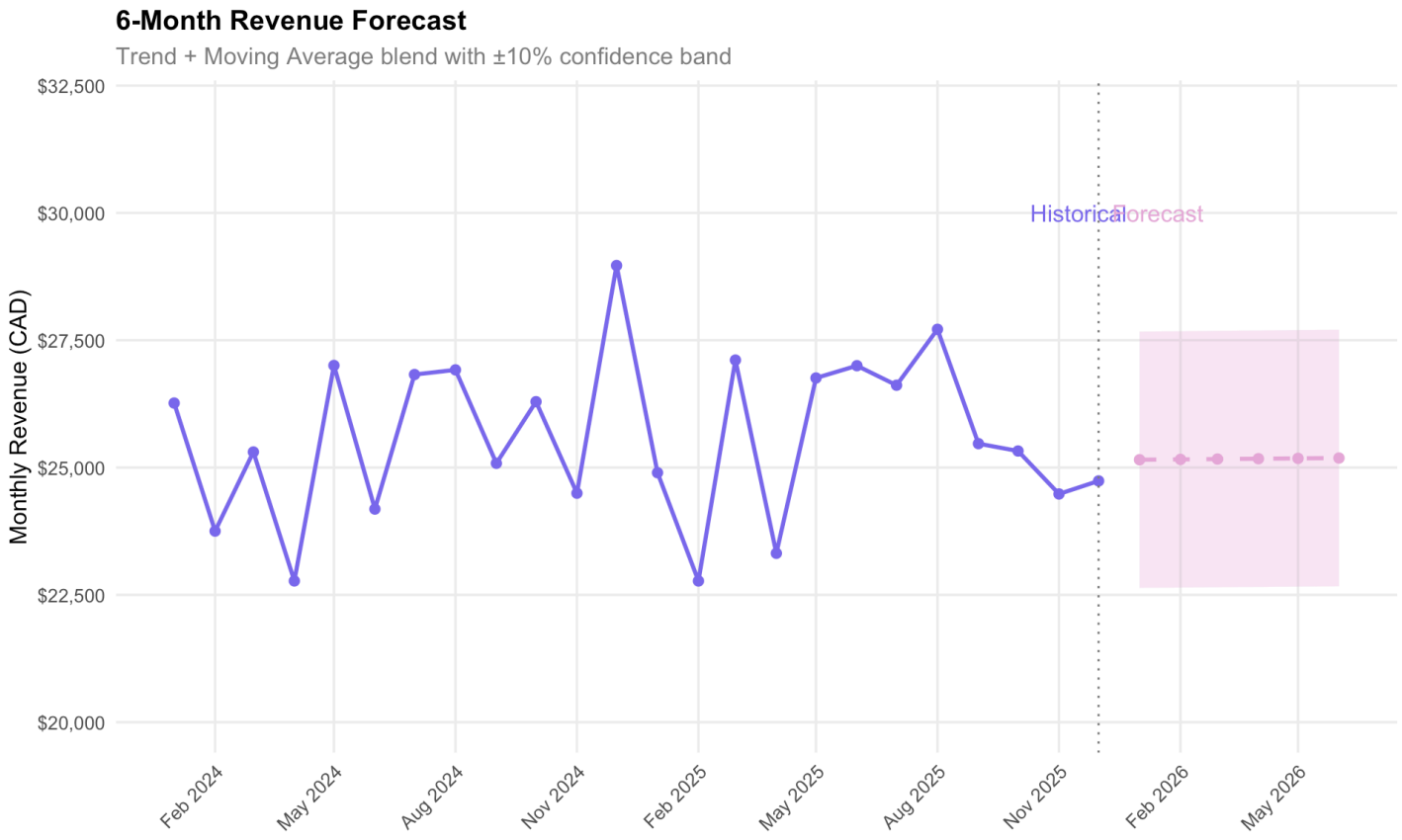
```
# 3-month moving average for smoothing
monthly_revenue <- monthly_revenue %>%
  mutate(
    ma_3 = zoo::rollmean(total_revenue, k = 3, fill = NA, align = "right"),
    trend_pred = predict(trend_model, .)
  )

# Forecast next 6 months
last_month <- max(monthly_revenue$month)
last_month_num <- max(monthly_revenue$month_num)
last_ma <- tail(na.omit(monthly_revenue$ma_3), 1)

# Simple exponential smoothing weight
alpha <- 0.3

forecast_df <- tibble(
  month = seq(last_month + months(1), by = "month", length.out = 6),
  month_num = (last_month_num + 1):(last_month_num + 6)
) %>%
  mutate(
    trend_forecast = predict(trend_model, newdata = .),
    # Blend trend with recent average (simple approach)
    blended_forecast = alpha * trend_forecast + (1 - alpha) * last_ma,
    forecast_low = blended_forecast * 0.90,
    forecast_high = blended_forecast * 1.10
  )

# Display forecast
forecast_df %>%
  mutate(
    month = format(month, "%b %Y"),
    blended_forecast = dollar(blended_forecast),
    forecast_low = dollar(forecast_low),
    forecast_high = dollar(forecast_high)
  ) %>%
  select(month, blended_forecast, forecast_low, forecast_high) %>%
  kable(
    col.names = c("Month", "Forecast", "Low (90%)", "High (110%)"),
    caption = "6-Month Revenue Forecast (CAD)"
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover"))
```

6-Month Revenue Forecast (CAD)

Month	Forecast	Low (90%)	High (110%)
Jan 2026	$25,153.69	$22,638.32	$27,669.06
Feb 2026	$25,160.41	$22,644.37	$27,676.45

Mar 2026	$25,167.13	$22,650.42	$27,683.85
Apr 2026	$25,173.86	$22,656.47	$27,691.24
May 2026	$25,180.58	$22,662.52	$27,698.64
Jun 2026	$25,187.30	$22,668.57	$27,706.03

```
# Combine historical and forecast data
all_data <- bind_rows(
  monthly_revenue %>%
    select(month, total_revenue) %>%
    mutate(type = "Historical"),
  forecast_df %>%
    select(month, blended_forecast, forecast_low, forecast_high) %>%
    rename(total_revenue = blended_forecast) %>%
    mutate(type = "Forecast")
)

# Plot
ggplot() +
  # Historical data
  geom_line(data = filter(all_data, type == "Historical"),
            aes(x = month, y = total_revenue), color = blume_colors[1], linewidth
= 1) +
  geom_point(data = filter(all_data, type == "Historical"),
             aes(x = month, y = total_revenue), color = blume_colors[1], size = 2)
+
  # Forecast with confidence band
  geom_ribbon(data = forecast_df,
              aes(x = month, ymin = forecast_low, ymax = forecast_high),
              fill = blume_colors[2], alpha = 0.3) +
  geom_line(data = filter(all_data, type == "Forecast"),
            aes(x = month, y = total_revenue), color = blume_colors[2], linewidth
= 1, linetype = "dashed") +
  geom_point(data = filter(all_data, type == "Forecast"),
             aes(x = month, y = total_revenue), color = blume_colors[2], size = 2)
+
  # Reference lines
  geom_vline(xintercept = last_month, linetype = "dotted", color = "gray50") +
  annotate("text", x = last_month - 15, y = 30000, label = "Historical", color = b
lume_colors[1]) +
  annotate("text", x = last_month + 45, y = 30000, label = "Forecast", color = blu
me_colors[2]) +
  scale_y_continuous(labels = dollar_format(), limits = c(20000, 32000)) +
  scale_x_date(date_labels = "%b %Y", date_breaks = "3 months") +
  labs(
    title = "6-Month Revenue Forecast",
    subtitle = "Trend + Moving Average blend with ±10% confidence band",
    x = NULL,
    y = "Monthly Revenue (CAD)"
  ) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Key Insight: Revenue appears relatively stable with gradual growth over time.

---

7. Executive Dashboard Summary

7.1 KPI Summary Table

Finally, I summarized the main metrics that a business team would likely track each month.

These KPIs help connect the analysis back to real operational decision-making.

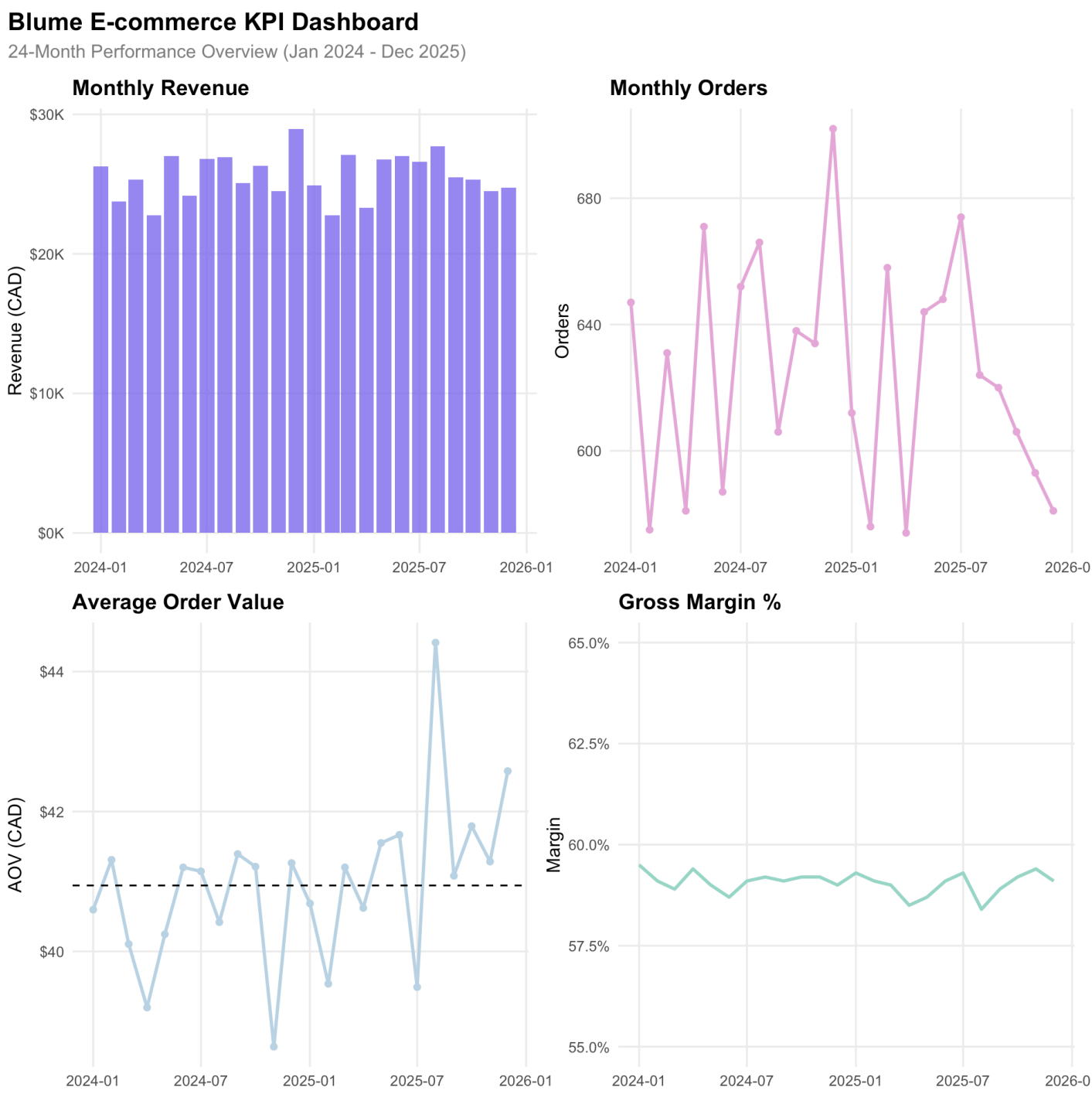
```
# Calculate overall KPIs
overall_kpis <- tibble(
  Metric = c("Total Revenue", "Total Orders", "Average Order Value",
             "Gross Margin %", "Monthly Churn Rate", "Repeat Customer Rate",
             "Subscription Revenue Share"),
  Value = c(
    dollar(sum(monthly_kpis$total_revenue)),
    comma(sum(monthly_kpis$orders)),
    dollar(mean(monthly_kpis$avg_order_value)),
    percent(mean(monthly_kpis$gross_margin_pct)/100, accuracy = 0.1),
    percent(mean(monthly_kpis$churn_rate), accuracy = 0.1),
    percent(repeat_rate, accuracy = 0.1),
    percent(sum(monthly_kpis$subscription_revenue) / sum(monthly_kpis$total_revenu
e), accuracy = 0.1)
  )
)

overall_kpis %>%
  kable(caption = "Executive KPI Summary (24 Months)") %>%
  kable_styling(bootstrap_options = c("striped", "hover"), full_width = FALSE) %>%
  row_spec(0, bold = TRUE, color = "white", background = blume_colors[1])
```

Executive KPI Summary (24 Months)

Metric	Value
Total Revenue	$614,078
Total Orders	15,000
Average Order Value	$40.94
Gross Margin %	59.1%
Monthly Churn Rate	7.0%
Repeat Customer Rate	96.7%
Subscription Revenue Share	18.4%

7.2 Monthly KPI Trend Dashboard

```
# Create a multi-panel dashboard
p1 <- ggplot(monthly_kpis, aes(x = month, y = total_revenue)) +
  geom_col(fill = blume_colors[1], alpha = 0.8) +
  scale_y_continuous(labels = dollar_format(scale = 0.001, suffix = "K")) +
  labs(title = "Monthly Revenue", x = NULL, y = "Revenue (CAD)")

p2 <- ggplot(monthly_kpis, aes(x = month, y = orders)) +
  geom_line(color = blume_colors[2], linewidth = 1) +
  geom_point(color = blume_colors[2]) +
  labs(title = "Monthly Orders", x = NULL, y = "Orders")

p3 <- ggplot(monthly_kpis, aes(x = month, y = avg_order_value)) +
  geom_line(color = blume_colors[3], linewidth = 1) +
  geom_point(color = blume_colors[3]) +
  geom_hline(yintercept = mean(monthly_kpis$avg_order_value), linetype = "dashed")
+
  scale_y_continuous(labels = dollar_format()) +
  labs(title = "Average Order Value", x = NULL, y = "AOV (CAD)")

p4 <- ggplot(monthly_kpis, aes(x = month, y = gross_margin_pct/100)) +
  geom_area(fill = blume_colors[4], alpha = 0.5) +
  geom_line(color = blume_colors[4], linewidth = 1) +
  scale_y_continuous(labels = percent_format(), limits = c(0.55, 0.65)) +
  labs(title = "Gross Margin %", x = NULL, y = "Margin")

# Combine plots
library(patchwork)
(p1 + p2) / (p3 + p4) +
  plot_annotation(
    title = "Blume E-commerce KPI Dashboard",
    subtitle = "24-Month Performance Overview (Jan 2024 - Dec 2025)",
    theme = theme(plot.title = element_text(face = "bold", size = 16))
  )
```



Key Insight: Overall, Blume's revenue is driven by a mix of best-selling single products and higher-value bundles, with subscriptions providing an additional recurring base.

Tracking AOV, churn, and gross margin together gives a clearer picture of growth and sustainability.

---

8. Key Findings & Recommendations

Summary of Insights

Area	Finding	Recommendation

Revenue Drivers	Bundle products drive 16% of revenue with high AOV ($79)	Promote bundle offerings; consider new bundle combinations
Pricing	59% of orders at full price; healthy margin preservation	Maintain strategic discount discipline
Profitability	Consistent 59% gross margin across categories	Monitor COGS; superfood lattes are margin leaders
Retention	7% monthly churn; critical window is months 1-3	Focus retention efforts on early subscriber engagement
Forecast	Stable ~$25K monthly revenue expected	Plan for consistent operations; no major seasonality

Limitations

This project uses simulated transaction data based on real Blume product pricing and general ecommerce benchmarks.

Because Blume's internal cost structure, marketing spend, and true customer behavior aren't publicly available, results should be viewed as educational rather than financial reporting.

Session Info

```
sessionInfo()
```

```
## R version 4.4.2 (2024-10-31)
## Platform: aarch64-apple-darwin20
## Running under: macOS Monterey 12.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRbl
as.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRla
pack.dylib;  LAPACK version 3.12.0
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/Vancouver
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] patchwork_1.3.2  kableExtra_1.4.0 scales_1.4.0     lubridate_1.9.5
##  [5] forcats_1.0.0    stringr_1.5.1    dplyr_1.1.4      purrr_1.0.4
##  [9] readr_2.1.5      tidyr_1.3.1      tibble_3.2.1     ggplot2_4.0.0
## [13] tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] sass_0.4.9         generics_0.1.3     xml2_1.3.7         lattice_0.22-6
##  [5] stringi_1.8.4      hms_1.1.3          digest_0.6.37      magrittr_2.0.3
##  [9] evaluate_1.0.3     grid_4.4.2         timechange_0.4.0   RColorBrewer_1.1-
3
## [13] fastmap_1.2.0      Matrix_1.7-1       jsonlite_1.9.1     mgcv_1.9-1
## [17] viridisLite_0.4.2  jquerylib_0.1.4    cli_3.6.4          rlang_1.1.5
## [21] crayon_1.5.3       splines_4.4.2      bit64_4.6.0-1      withr_3.0.2
## [25] cachem_1.1.0       yaml_2.3.10        tools_4.4.2        parallel_4.4.2
## [29] tzdb_0.4.0         vctrs_0.6.5        R6_2.6.1           zoo_1.8-15
## [33] lifecycle_1.0.4    bit_4.6.0          vroom_1.6.5        pkgconfig_2.0.3
## [37] pillar_1.10.1      bslib_0.9.0        gtable_0.3.6       glue_1.8.0
## [41] systemfonts_1.2.1  xfun_0.56          tidyselect_1.2.1   rstudioapi_0.17.1
## [45] knitr_1.51         farver_2.1.2       nlme_3.1-166       htmltools_0.5.8.1
## [49] labeling_0.4.3     rmarkdown_2.30     svglite_2.1.3      compiler_4.4.2
## [53] S7_0.2.0
```