Final Report

Sheamin Kim

2025-03-18

```
# Loading necessary libraries
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3
```

```
## Warning: package 'readr' was built under R version 4.1.3
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

```
## Warning: package 'stringr' was built under R version 4.1.3
```

```
## Warning: package 'forcats' was built under R version 4.1.3
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.1.3
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.1.3
```

Background and Motivation

PASTE HERE

Data Cleaning/Prep

```
## PRICE DATA CLEANING

vax_df <- read.csv("cdc_vaccine_prices_full.csv")
inflation_df <- read.csv("inflation_cpis.csv")

#standardizing values, getting rid of $
vax_df$Private.Sector.Cost..Dose = gsub("\\$", "", vax_df$Private.Sector.Cost..Dose)
vax_df$CDC.Cost..Dose = gsub("\\$", "", vax_df$CDC.Cost..Dose)
```

```r
vax_df$Private.Sector.Cost..Dose <- as.numeric(as.character(vax_df$Private.Sector.Cost..Dose))

vax_df$CDC.Cost..Dose <- as.numeric(as.character(vax_df$CDC.Cost..Dose))

vax_df$Date <- as.Date(vax_df$Date)


## Adding inflation data
# extract the year and convert to numeric format
vax_df$year <- as.numeric(format(vax_df$Date, "%Y"))

vax_df = merge(x = vax_df, y = inflation_df, by = "year")
vax_df$CPI <- as.numeric(as.character(vax_df$CPI))
sapply(vax_df, class)
```

```
##                       year                    Vaccine      Brandname..Tradename
##                  "numeric"                "character"               "character"
##                        NDC                  Packaging            CDC.Cost..Dose
##                "character"                "character"                 "numeric"
## Private.Sector.Cost..Dose          Contract.End.Date              Manufacturer
##                  "numeric"                "character"               "character"
##            Contract.Number                       Date                  ï..place
##                "character"                     "Date"                 "logical"
##                        CPI             percent.change
##                  "numeric"                "character"
```

```r
reference_year <- 2009

# Get CPI for the reference year
reference_cpi <- vax_df$CPI[vax_df$year == reference_year]

# the type of below should be double
# print(typeof(reference_cpi))

# Adjust prices for inflation based on the reference CPI
vax_df$adjusted_price <- vax_df$Private.Sector.Cost..Dose * (reference_cpi / vax_df$CPI)
```

```
## Warning in reference_cpi/vax_df$CPI: longer object length is not a multiple of
## shorter object length
```

```r
vax_df$adjusted_price_cdc <- vax_df$CDC.Cost..Dose * (reference_cpi / vax_df$CPI)
```

```
## Warning in reference_cpi/vax_df$CPI: longer object length is not a multiple of
## shorter object length
```

```r
## RATES DATA CLEANING
df1 <- read.csv("monthly_cumulative.csv")

# Define the correct month order
month_levels <- c("SEP","OCT","NOV","DEC","JAN","FEB","MAR","APR","MAY","JUN","JUL","AUG")
```

```r
# Step 1: Ensure month is a factor for proper sorting
df1 <- df1 %>%
  mutate(month = factor(month, levels = month_levels))

# getting new dose numbers
rates_df <- df1 %>%
  arrange(current_season, jurisdiction, age_group_label, month) %>%
  group_by(current_season, jurisdiction, age_group_label) %>%
  mutate(
    new_doses = numerator - lag(numerator, default = NA)  # New doses = current - previous
  ) %>%
  ungroup()

overall_pop <- rates_df %>%
  filter(age_group_label == "Overall") %>%
  select(jurisdiction, current_season, population) %>%
  rename(overall_population = population)

rates_df <- rates_df %>%
  left_join(overall_pop, by = c("jurisdiction", "current_season"))

rates_df <- rates_df %>%
  mutate(vax_rate = new_doses / coalesce(population, overall_population))

rates_df <- rates_df %>%
  mutate(start_year = as.integer(substr(current_season, 1, 4)),  # Extract the first year
         # Combine starting year with month to create a valid date
         date = as.Date(paste(start_year, month, "01", sep = "-"), format = "%Y-%b-%d"))

df_dedup <- rates_df %>%
  group_by(jurisdiction, age_group_label, current_season, date) %>%
  slice(1) %>%               # Keep only the first row for each group
  ungroup()

# 2. Sort by jurisdiction, age group, season, and date to ensure proper order for calculating new doses
df_dedup <- df_dedup %>%
  arrange(jurisdiction, age_group_label, current_season, date)

# 3. Calculate new doses by comparing the cumulative totals
df_dedup <- df_dedup %>%
  group_by(jurisdiction, age_group_label, current_season) %>%
  mutate(new_doses = numerator - lag(numerator)) %>%    # Subtract previous month from current month
  ungroup() %>%
  mutate(new_doses = ifelse(new_doses < 0, 0, new_doses))


## ED VISITS DATA CLEANING
file1 <- "ed_traj.csv"
file2 <- "ed_visits.csv"
df_1 <- read.csv(file1)
df_2 <- read.csv(file2)
# Convert week_end to Date format
df_1$week_end <- as.Date(df_1$week_end, format="%Y-%m-%d")
```

```r
df_2$week_end <- as.Date(df_2$week_end, format="%Y-%m-%d")

# Clean df1 (Trajectories dataset) - Select relevant columns
df1_clean <- df_1 %>%
  select(week_end, geography, county, percent_visits_influenza) %>%
  filter(!is.na(percent_visits_influenza))
# Clean df2 (Demographics dataset) - Select flu data only
df2_clean <- df_2 %>%
  filter(pathogen == "Influenza") %>% # Select only Influenza-related ED visits
  select(week_end, geography, percent_visits) %>%
  rename(percent_visits_influenza = percent_visits) %>%
  filter(!is.na(percent_visits_influenza))

# Merge both datasets for better insights
df_combined <- bind_rows(df1_clean, df2_clean)

df_combined$Date <- as.Date(df_combined$week_end)

df_combined$year <- format(df_combined$week_end, "%Y")
df_combined$month <- format(df_combined$week_end, "%m")
df_combined$month_abbr <- month.abb[as.numeric(df_combined$month)]

seasonal <- df_combined %>%
  filter(county == "All") %>%
  group_by(Date) %>%
  summarise(percent_visits_influenza = mean(percent_visits_influenza))

seasonal$Date <- as.Date(seasonal$Date)
seasonal$year <- format(seasonal$Date, "%Y")
seasonal$month <- format(seasonal$Date, "%m")
seasonal$month_abbr <- month.abb[as.numeric(seasonal$month)]
```

Datasets

1. Flu vaccination rates

- https://healthdata.gov/dataset/Monthly-Cumulative-Number-and-Percent-of-Persons-W/8y48-wjrp/about_data

The flu vaccination rates dataset, sourced from HealthData.gov and maintained by the CDC, provides monthly cumulative counts and percentages of individuals who have received at least one dose of the influenza vaccine. The data spans multiple flu seasons, from 2019 to 2023, and is categorized by age group and jurisdiction (states, territories, and select cities). The data set is compiled from Immunization Information Systems (IIS), which aggregate vaccine administration data from various public health agencies.

This data set offers insights into vaccination trends over time and across different demographic groups. The cumulative nature of the records ensures that historical data is preserved, allowing for trend analysis. However, the data set has limitations, including variations in data completeness across jurisdictions and differences in state policies regarding vaccine data reporting. The population denominators used for calculating vaccination rates are sourced from the U.S. Census Bureau's 2020 estimates. Standard errors are not provided, as the data includes all vaccinations rather than a sample.

[EXPLANATION FOR THESE HERE]

```r
## Summary Table
monthly_trends <- rates_df %>%
  group_by(current_season, month) %>%
  summarise(new_doses = sum(new_doses, na.rm = TRUE)) %>%
  arrange(current_season, month)
```

```
## `summarise()` has grouped output by 'current_season'. You can override using
## the `.groups` argument.
```

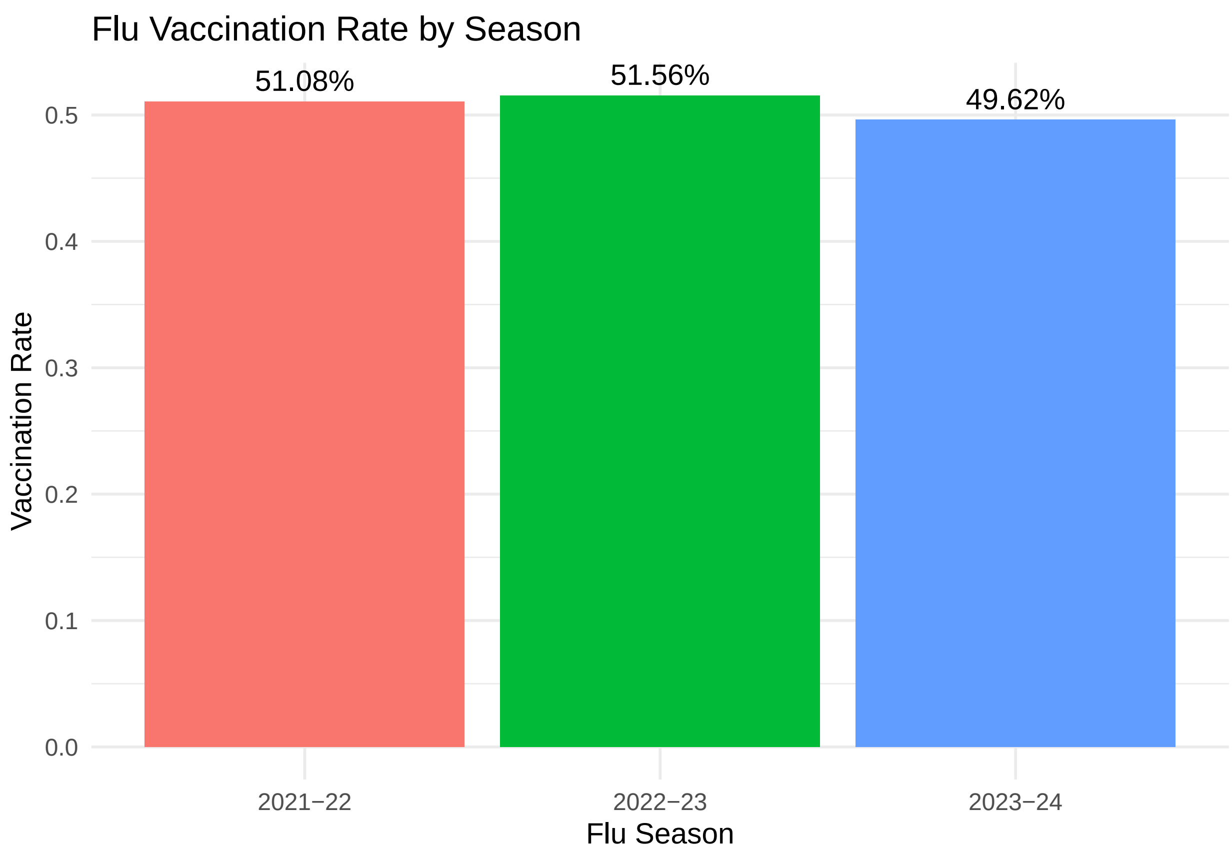
```r
print(monthly_trends)
```

```
## # A tibble: 33 x 3
## # Groups:   current_season [3]
##    current_season month  new_doses
##    <chr>          <fct>      <dbl>
##  1 2021-22        SEP            0
##  2 2021-22        OCT   1059540168
##  3 2021-22        NOV    564210528
##  4 2021-22        DEC    290056452
##  5 2021-22        JAN    135807960
##  6 2021-22        FEB     64187604
##  7 2021-22        MAR     37405272
##  8 2021-22        APR     14005248
##  9 2021-22        MAY      6891072
## 10 2021-22        JUN      3407448
## # ... with 23 more rows
```

```r
## Basic Bar Chart
df_clean <- df1 %>%
  filter(!is.na(numerator)) %>% # Remove rows where numerator is NA
  arrange(jurisdiction, current_season, age_group_label, month) %>% # Sort by jurisdiction, season, and
  group_by(jurisdiction, current_season, age_group_label) %>% # Group by jurisdiction, season, and age
  mutate(monthly_doses = numerator - lag(numerator)) %>% # Subtract previous month from current to get
  ungroup() %>% # Remove the grouping
  filter(!is.na(monthly_doses) & monthly_doses >= 0) # Remove NAs and negative values (in case of any

population_per_season_jurisdiction <- df_clean %>%
  filter(age_group_label == "Overall") %>% # Only include rows where age_group_label is "Overall"
  group_by(current_season, jurisdiction) %>%
  summarise(
    unique_population = unique(population), # Get a single unique population value per jurisdiction
    total_doses = sum(monthly_doses, na.rm = TRUE)) %>%
  group_by(current_season) %>%
  summarise(
    total_population = sum(unique_population, na.rm = TRUE), # Sum up the unique population across all
    total_doses = sum(total_doses, na.rm = TRUE)) %>%
  mutate(vaccination_rate = (total_doses / total_population) * 2.5)
```

```
## `summarise()` has grouped output by 'current_season'. You can override using
## the `.groups` argument.
```

```r
population_per_season_jurisdiction %>%
  ggplot(aes(x = current_season, y = vaccination_rate, fill = current_season)) +
  geom_col() +
  geom_text(aes(label = scales::percent(vaccination_rate)), vjust = -0.5, size = 4) +
  labs(title = "Flu Vaccination Rate by Season",
       x = "Flu Season",
       y = "Vaccination Rate") +
  theme_minimal() +
  theme(legend.position = "none")
```



2. Flu vaccination expenditure

- https://www.cdc.gov/vaccines-for-children/php/awardees/current-cdc-vaccine-price-list.html
- https://www.cdc.gov/vaccines/programs/vfc/awardees/vaccine-management/price-list/archive.html

The flu vaccination expenditure dataset is derived from CDC's Vaccine Price Lists, which detail both public-sector contract prices and private-sector prices for influenza vaccines. The dataset includes pricing information for pediatric and adult flu vaccines, with historical records dating back to 2001. The primary source for current vaccine prices is the CDC's publicly available vaccine price list, while archived prices are stored separately.

The dataset includes details such as vaccine brand names, National Drug Codes (NDCs), packaging information, CDC cost per dose, private sector cost per dose, contract end dates, and manufacturers. This data allows for an analysis of pricing trends over time, identifying fluctuations in vaccine costs and potential disparities between public and private sector pricing. However, obtaining historical data required web scraping or API access, as the archived prices are distributed across multiple web pages.

[EXPLANATION HERE]

```r
## Summary Table
summary_table <- vax_df %>%
  group_by(year) %>%
  summarise(
    num_products = n(),
    avg_cdc_price = mean(CDC.Cost..Dose, na.rm = TRUE),
    avg_private_price = mean(Private.Sector.Cost..Dose, na.rm = TRUE),
    avg_adj_cdc_price = mean(adjusted_price_cdc, na.rm = TRUE),
    avg_adj_private_price = mean(adjusted_price, na.rm = TRUE),
    min_private_price = min(Private.Sector.Cost..Dose, na.rm = TRUE),
    max_private_price = max(Private.Sector.Cost..Dose, na.rm = TRUE),
  )

print(summary_table)
```

```
## # A tibble: 16 x 8
##     year num_products avg_cdc_price avg_private_price avg_adj_cdc_price
##    <dbl>        <int>         <dbl>             <dbl>             <dbl>
##  1  2009           32          7.99              11.2              7.99
##  2  2010           32          9.78              11.9              9.62
##  3  2011           37         10.5               12.2              9.99
##  4  2012           39          9.23              12.2              8.63
##  5  2013           43          8.77              12.8              8.08
##  6  2014           32          9.14              13.8              8.28
##  7  2015           44         10.5               16.0              9.49
##  8  2016           35         11.8               17.9             10.6
##  9  2017           32         12.2               17.6             10.6
## 10  2018           26         12.4               17.6             10.6
## 11  2019           27         12.8               18.1             10.7
## 12  2020           31         13.4               19.5             11.1
## 13  2021           32         13.9               19.8             11.0
## 14  2022           32         14.5               20.4             10.7
## 15  2023           32         15.1               21.2             10.7
## 16  2024           24         15.8               23.1             10.8
## # ... with 3 more variables: avg_adj_private_price <dbl>,
## #   min_private_price <dbl>, max_private_price <dbl>
```

3. Flu emergency department visit rates

- https://healthdata.gov/dataset/NSSP-Emergency-Department-Visit-Trajectories-by-St/hr4c-e7p6/about_data
- https://healthdata.gov/dataset/NSSP-Emergency-Department-Visits-COVID-19-Flu-RSV-/vfw5-fbqw/about_data

The flu emergency department (ED) visit rates dataset is sourced from the National Syndromic Surveillance Program (NSSP) and published on HealthData.gov. This dataset provides the percentage of emergency department visits that are attributed to influenza, alongside data for other respiratory illnesses such as COVID-19 and RSV. The dataset spans from 2022 to 2025 and is updated weekly.

The data set is available in two formats:

- **NSSP Emergency Department Visit Trajectories by State and Sub-State Regions**: This data set reports the percentage of ED visits for flu at both state and sub-state (Health Service Area) levels. It also includes trend classifications (increasing, decreasing, or stable) based on statistical models.
- **NSSP Emergency Department Visits by Demographic Category**: This dataset categorizes ED visits for influenza by demographic variables such as age, sex, and race/ethnicity. It provides insights into disparities in flu-related ED visits across different population groups.

The data is collected from health facilities participating in the NSSP and is intended to track trends over time.

[EXPLANATION HERE]

```r
# Create yearly summary table (limit to top 10 states)
summary_table <- df_combined %>%
  mutate(year = year(week_end)) %>%
  group_by(year, geography) %>%
  summarize(avg_percent_influenza = mean(percent_visits_influenza, na.rm = TRUE), .groups = "drop") %>%
  arrange(desc(avg_percent_influenza)) %>%
  group_by(year) %>%
  slice_max(order_by = avg_percent_influenza, n = 10) # Keep only the top 10 states

print(summary_table)
```

```
## # A tibble: 40 x 3
## # Groups:   year [4]
##     year geography      avg_percent_influenza
##    <dbl> <chr>                          <dbl>
##  1  2022 Mississippi                     5.66
##  2  2022 New Mexico                      5.23
##  3  2022 Alabama                         5.22
##  4  2022 Kentucky                        4.99
##  5  2022 North Carolina                  4.93
##  6  2022 Indiana                         4.92
##  7  2022 Virginia                        4.85
##  8  2022 South Carolina                  4.74
##  9  2022 Texas                           4.61
## 10  2022 West Virginia                   4.44
## # ... with 30 more rows
```

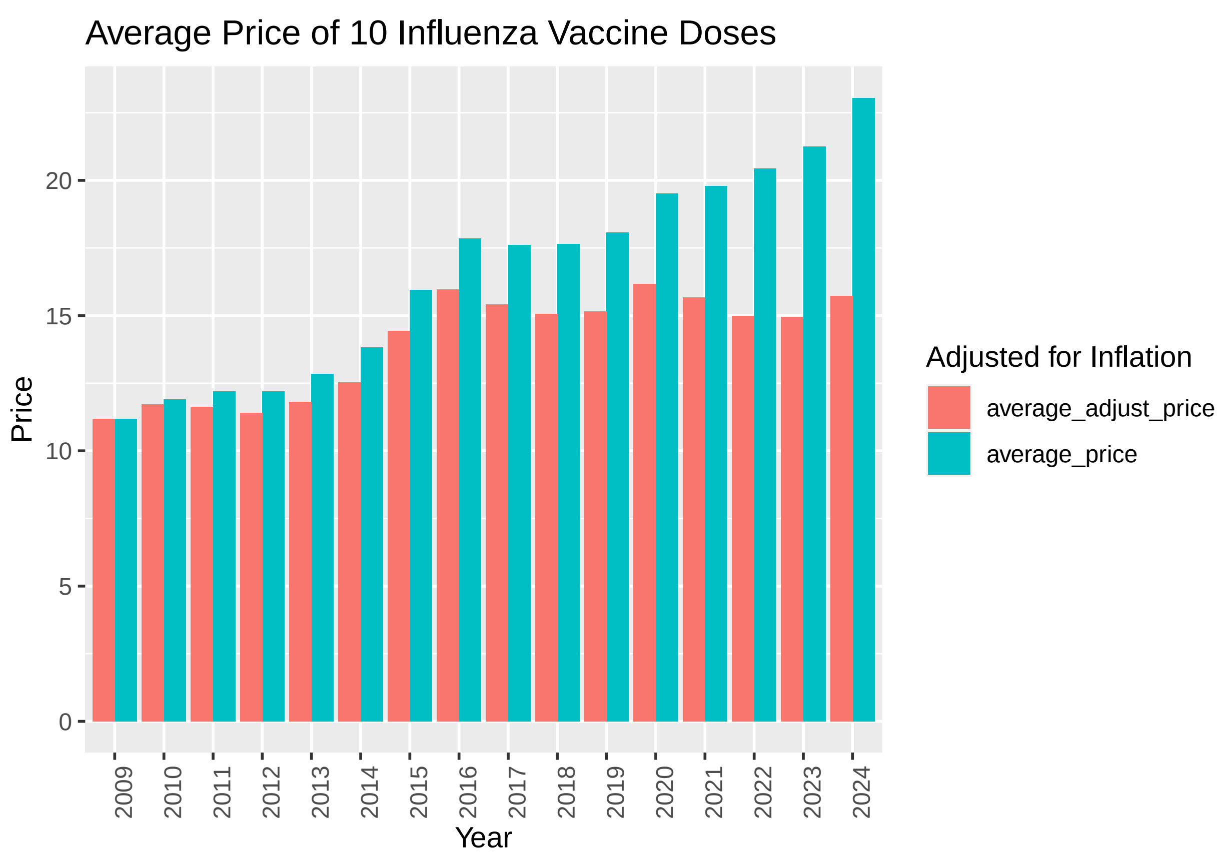
Methods

TEXT HERE

Results

Individual Analysis

```r
## EDIT TITLE
plot_1 <- vax_df %>%
  group_by(year) %>%
  summarise(average_price = mean(Private.Sector.Cost..Dose),
            average_adjust_price = mean(adjusted_price)) %>%
  pivot_longer(cols = c("average_price", "average_adjust_price"),
               names_to = "price_type",
               values_to = "price") %>%
  ggplot(aes(x=factor(year), y=price, fill=price_type)) +
  geom_col(position="dodge") +
  theme(axis.text.x = element_text(angle = 90)) +
  scale_x_discrete(labels = 2009:2025, breaks = 2009:2025) +
  labs(title = "Average Price of 10 Influenza Vaccine Doses",
       x = "Year",
       y = "Price",
       fill = "Adjusted for Inflation")

print(plot_1)
```

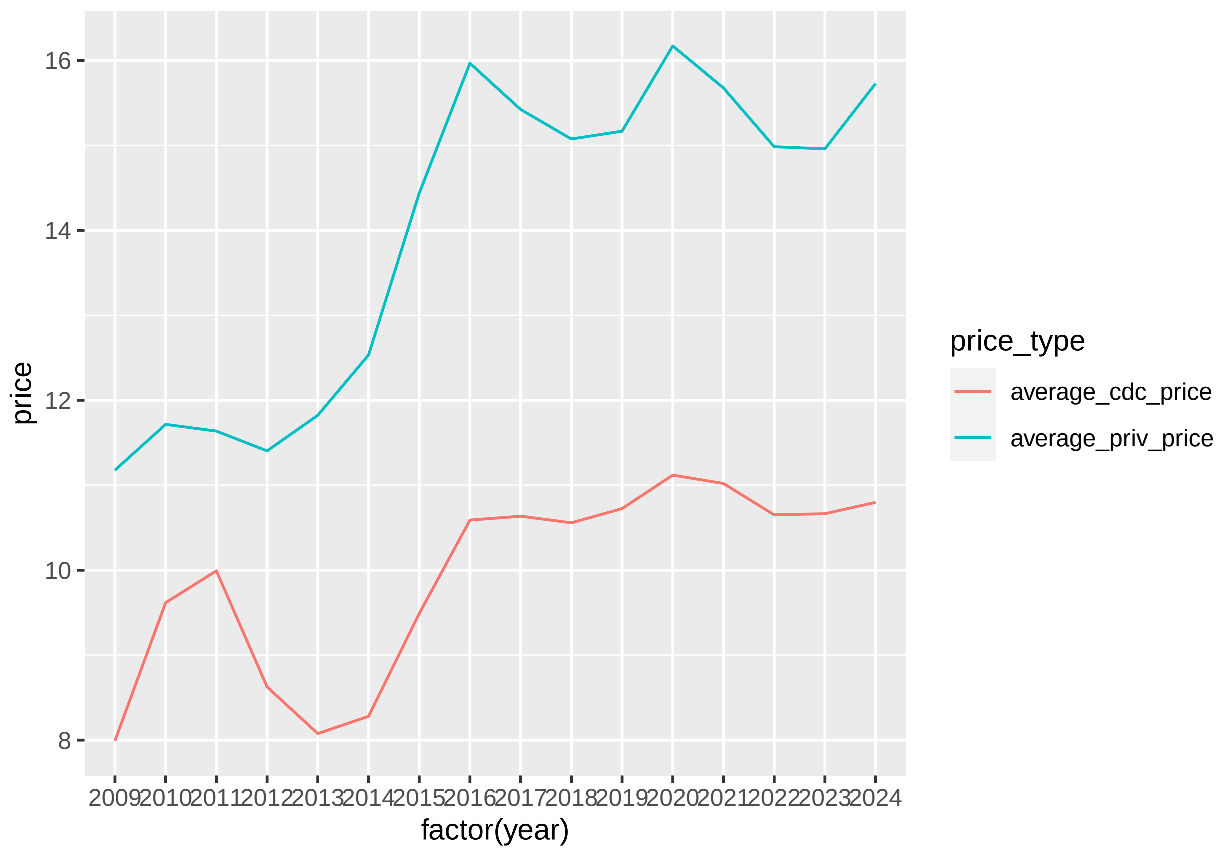


Price Data

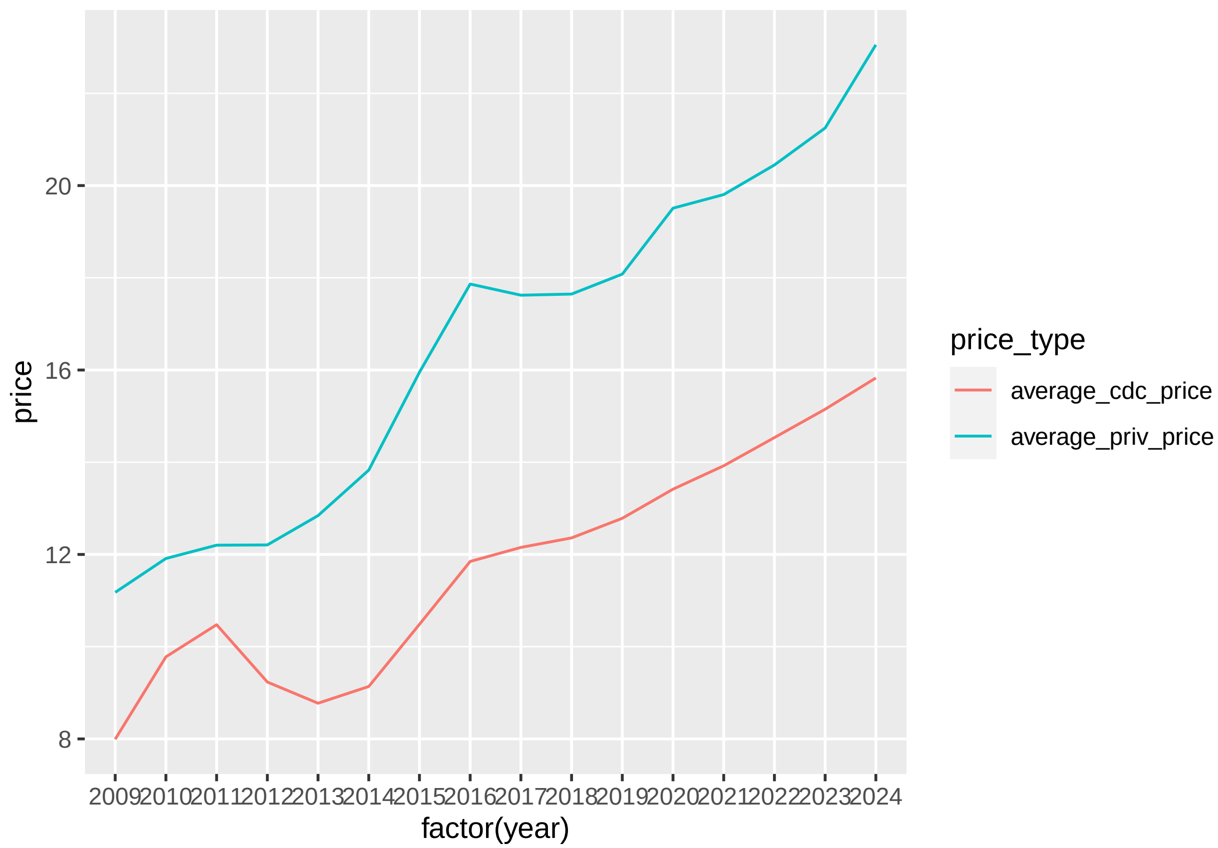
EXPLAIN HERE

```r
##EDIT TITLES AND STUFF
adj_private_cdc_comparison_plot <- vax_df %>%
  group_by(year) %>%
```

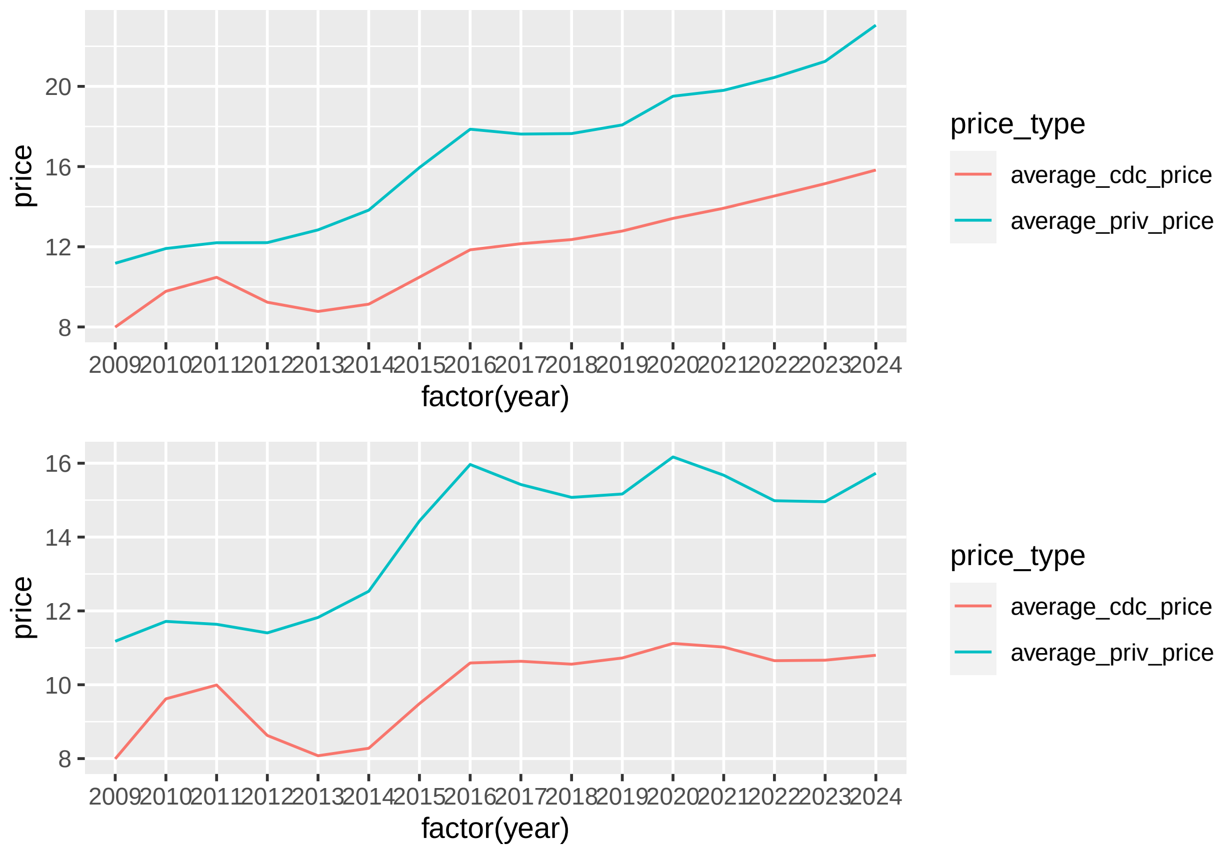
```r
  summarise(average_priv_price = mean(adjusted_price),
            average_cdc_price = mean(adjusted_price_cdc)) %>%
  pivot_longer(cols = c("average_priv_price", "average_cdc_price"),
               names_to = "price_type",
               values_to = "price") %>%
  ggplot(aes(x=factor(year), y=price, group=price_type)) +
  geom_line(aes(color=price_type))

adj_private_cdc_comparison_plot
```




```r
private_cdc_comparison_plot <- vax_df %>%
  group_by(year) %>%
  summarise(average_priv_price = mean(Private.Sector.Cost..Dose),
            average_cdc_price = mean(CDC.Cost..Dose)) %>%
  pivot_longer(cols = c("average_priv_price", "average_cdc_price"),
               names_to = "price_type",
               values_to = "price") %>%
  ggplot(aes(x=factor(year), y=price, group=price_type)) +
  geom_line(aes(color=price_type))

private_cdc_comparison_plot
```




```r
print(grid.arrange(private_cdc_comparison_plot, adj_private_cdc_comparison_plot))
```




```
## TableGrob (2 x 1) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (2-2,1-1) arrange gtable[layout]
```

EXPLAIN ABOVE PLOT HERE

```
test <- t.test(vax_df$Private.Sector.Cost..Dose, vax_df$CDC.Cost..Dose,
               alternative = "greater")
test
```

```
##
##  Welch Two Sample t-test
##
## data:  vax_df$Private.Sector.Cost..Dose and vax_df$CDC.Cost..Dose
## t = 17.525, df = 943.6, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  4.302051      Inf
## sample estimates:
## mean of x mean of y
##  16.27383  11.52568
```

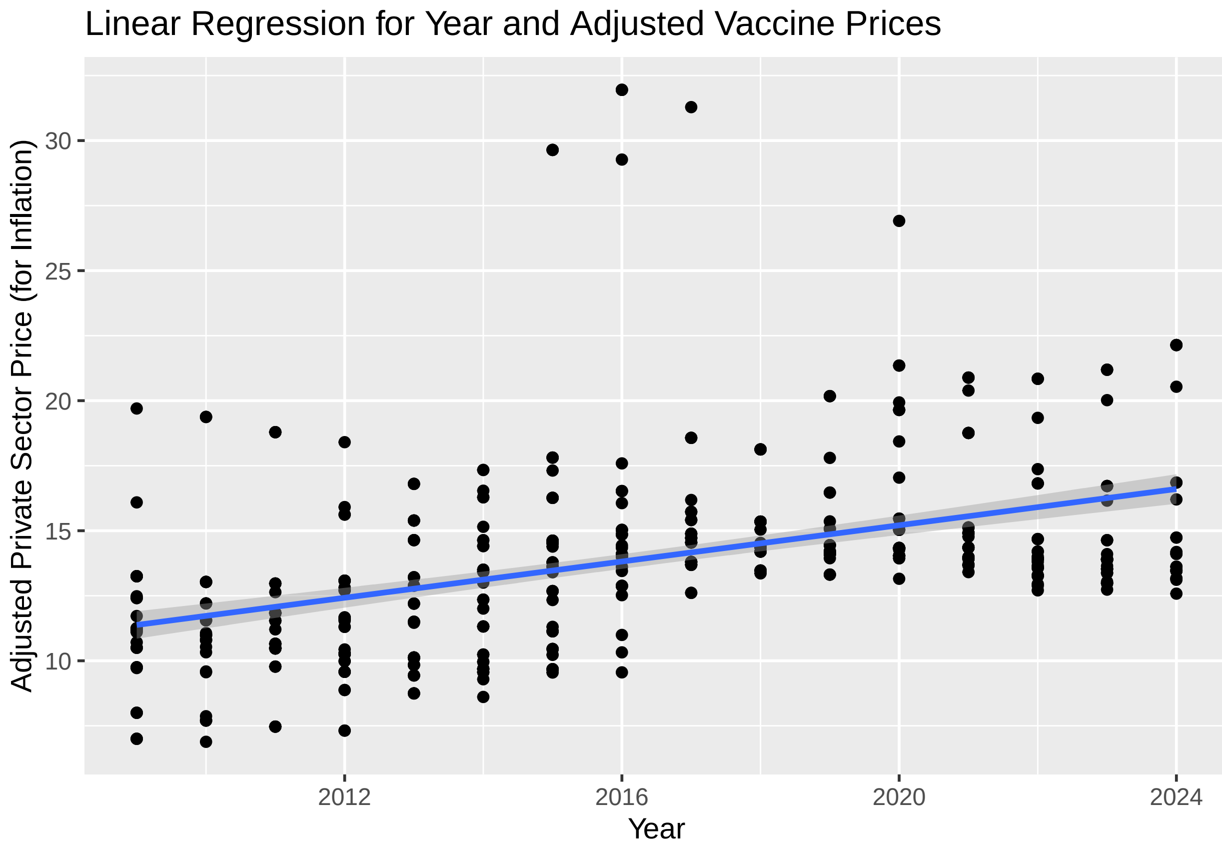
EXPLAIN T-TEST RESULTS HERE

```r
# lin reg - is there a correlation between year and vaccine price
model <- lm(adjusted_price ~ year, data = vax_df)
print(summary(model))
```

```
##
## Call:
## lm(formula = adjusted_price ~ year, data = vax_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.1079 -2.0896 -0.6466  1.0323 18.1349
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -688.58851   64.19093  -10.73   <2e-16 ***
## year           0.34842    0.03184   10.94   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.301 on 528 degrees of freedom
## Multiple R-squared:  0.1849, Adjusted R-squared:  0.1833
## F-statistic: 119.8 on 1 and 528 DF,  p-value: < 2.2e-16
```

```r
ggplot(vax_df, aes(x = year, y = adjusted_price)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Linear Regression for Year and Adjusted Vaccine Prices",
       x = "Year",
       y = "Adjusted Private Sector Price (for Inflation)")
```

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

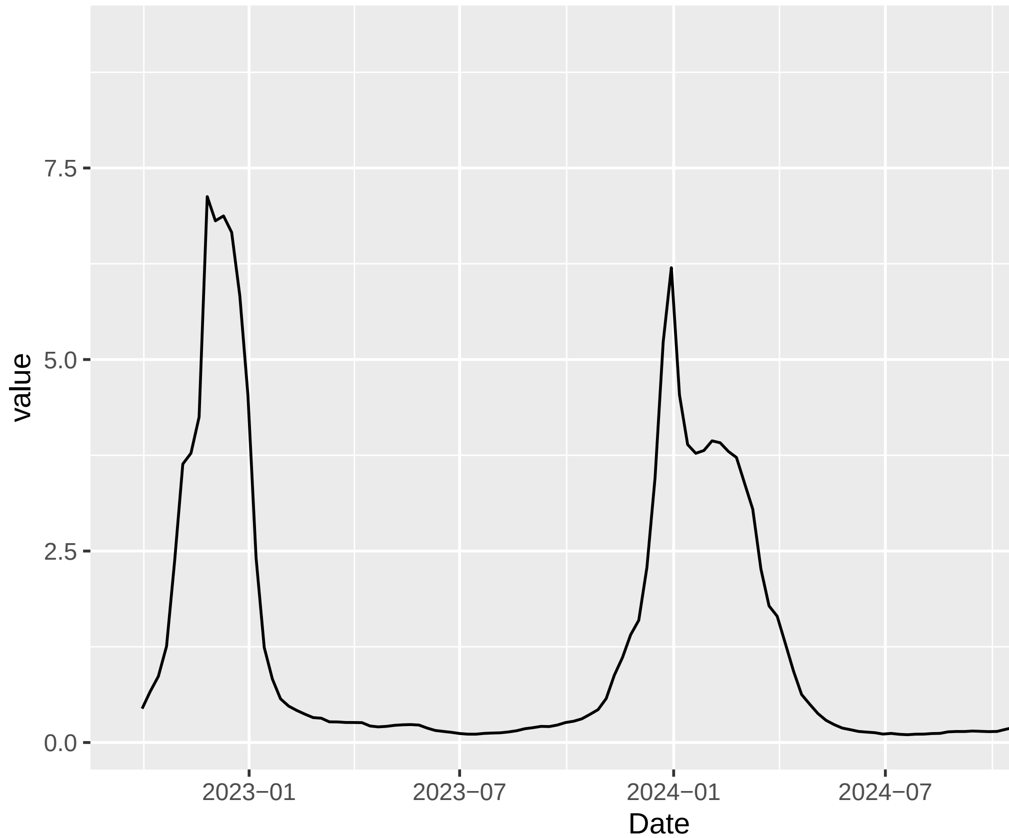


EXPLAIN PLOT HERE

```
## ANOVA CODE HERE
```

Vaccination Rates EXPLAIN OUTPUT HERE

```
## MAKE THIS PRETTIER
df_for_plot <- df_combined %>%
  group_by(Date) %>%
  summarise(value = mean(percent_visits_influenza))

plot_time <- ggplot(data = df_for_plot) +
  geom_line(aes(x = Date, y = value))

plot_time
```



Emergency Department Visits

EXPLAIN PLOT HERE

```
anova_result <- aov(percent_visits_influenza ~ as.factor(year), data = df_combined)
summary(anova_result) # Print ANOVA test result
```

```
##                     Df  Sum Sq Mean Sq F value Pr(>F)
## as.factor(year)      3  733391  244464   39752 <2e-16 ***
## Residuals       294524 1811259       6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

EXPLAIN FINDINGS

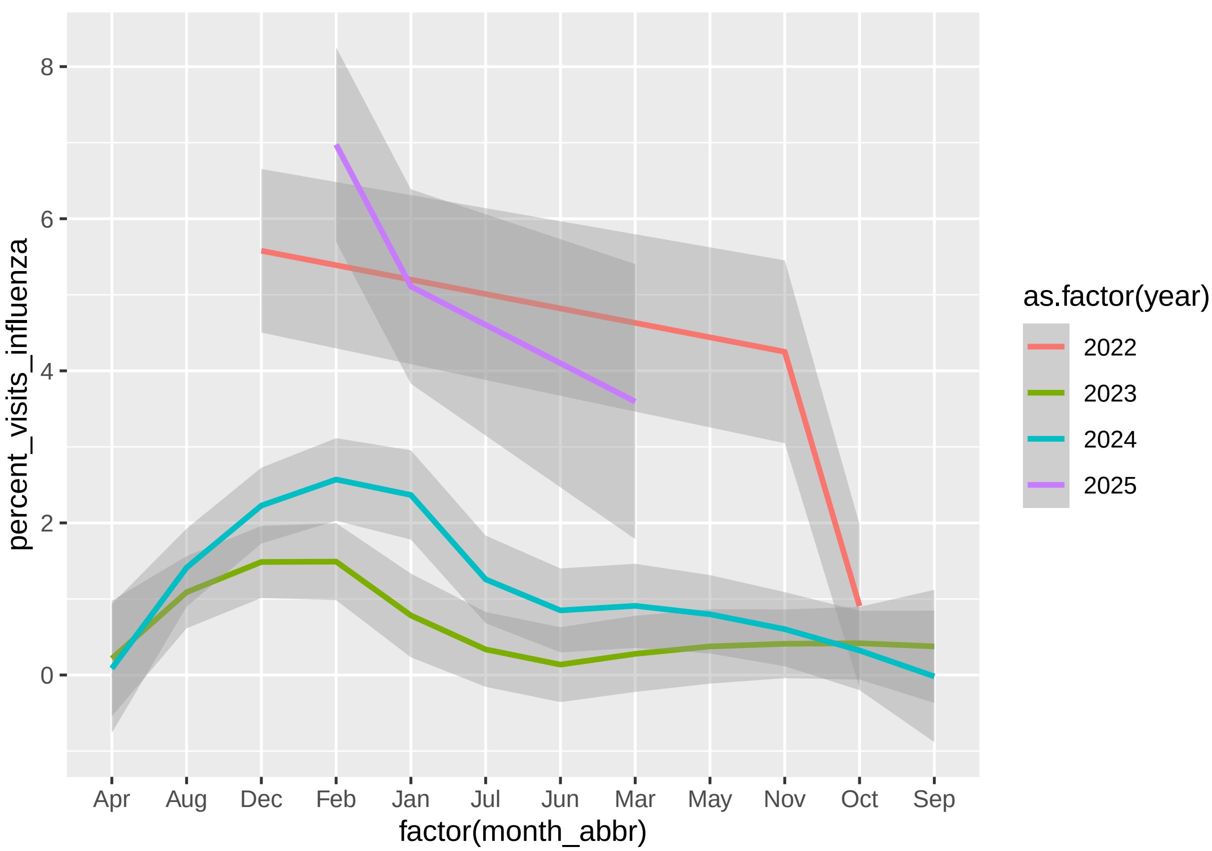
Relationship Analysis

```
df_for_cop <- df_dedup %>%
  group_by(date) %>%
  summarise(plot_col = sum(new_doses, na.rm = TRUE))

## MAKE THIS PRETTIER
ggplot(seasonal, aes(x = factor(month_abbr), y = percent_visits_influenza, color = as.factor(year), grou
  geom_smooth()
```

Vaccination Rates and Emergency Department Visits

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

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 2.96

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 8.04

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 1.6386e-016

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 64.642

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at
## 2.96

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 8.04

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 1.6386e-016

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 64.642

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 3.98

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 1.02

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 16.16

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used at
## 3.98
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius 1.02

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal condition
## number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other near
## singularities as well. 16.16
```




```r
#ggplot() +
 # geom_line(data=seasonal, aes(x = Date, y = percent_visits_influenza)) +
  #geom_line(data=temp_df, aes(x = date, y = avg_rate))

merged = merge(df_for_cop, seasonal, by.x="date", by.y="Date")

## BAD TEST HERE
# cor.test(merged$plot_col, merged$percent_visits_influenza, method = "pearson")
```

text here explaining

```r
all_totals <- df_dedup %>%
  group_by(date) %>%
  summarise(total_doses = sum(new_doses, na.rm = TRUE))

all_totals$year <- format(all_totals$date, "%Y")

all_totals$doses_div_ten <- (all_totals$total_doses) / 10

yr_totals <- all_totals %>%
  group_by(year) %>%
  summarise(total_doses = sum(total_doses))


price_table <- vax_df %>%
  group_by(year) %>%
  summarise(cost = mean(adjusted_price))

price_table$cost_per_dose <- (price_table$cost) / 10

price_yr_totals <- merge(x = yr_totals, y = price_table, by = "year")

price_yr_totals$money_spent <- price_yr_totals$total_doses * price_yr_totals$cost_per_dose

print(price_yr_totals)
```

Vaccination Rates and Vaccine Product Prices

```
##   year total_doses     cost cost_per_dose money_spent
## 1 2021   219844791 15.67542      1.567542   344615912
## 2 2022   224176457 14.98390      1.498390   335903771
## 3 2023   208979305 14.95873      1.495873   312606423
```

SOME TEXT HERE

```r
anova_result <- aov(money_spent ~ as.factor(year), data = price_yr_totals)
summary(anova_result)
```

```
##                 Df    Sum Sq   Mean Sq
## as.factor(year)  2 5.478e+14 2.739e+14
```

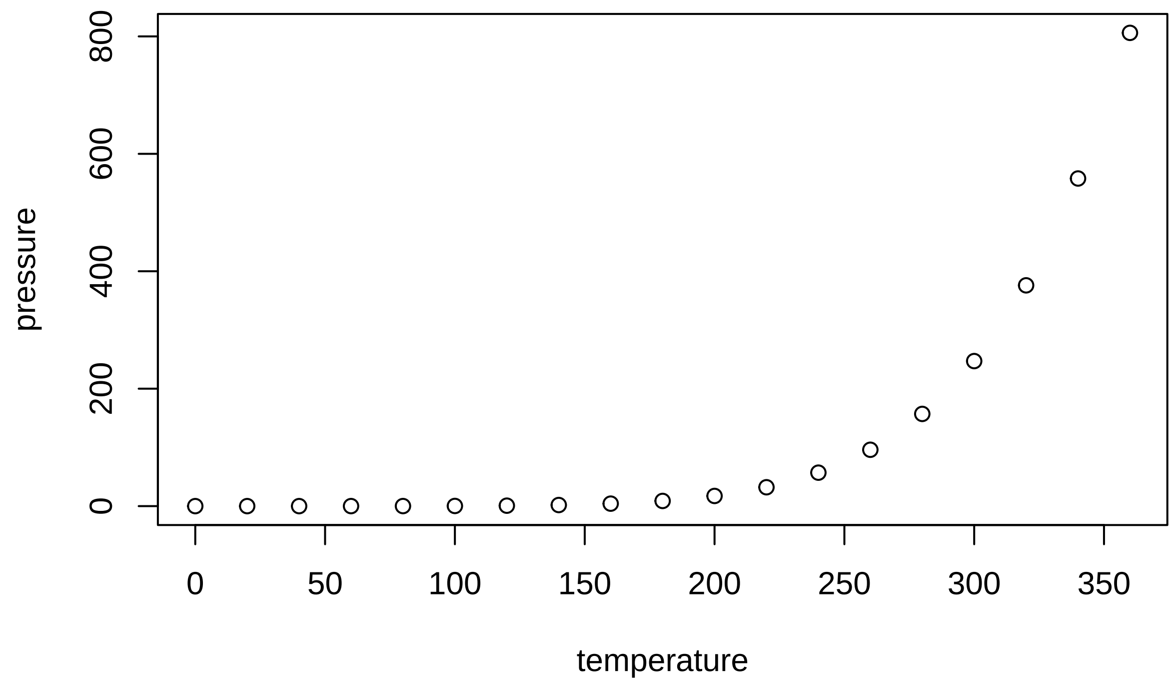
EXPLANATION HERE

Discussion

text here

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.