STATS_451_final_eda

Sirui Chen, Yingxi Chen, Jiayu Feng, Qiyun Teng

2024-04-15

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## Warning: package 'tibble' was built under R version 4.3.3
```

```
## Warning: package 'tidyr' was built under R version 4.3.3
```

```
## Warning: package 'readr' was built under R version 4.3.3
```

```
## Warning: package 'purrr' was built under R version 4.3.3
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
## Warning: package 'forcats' was built under R version 4.3.3
```

```
## Warning: package 'lubridate' was built under R version 4.3.3
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2     
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(readr)
library(ggplot2)
library(rstan)
```

```
## Warning: package 'rstan' was built under R version 4.3.3
```

```
## Loading required package: StanHeaders
```

```
## Warning: package 'StanHeaders' was built under R version 4.3.3
```

```
##
## rstan version 2.32.6 (Stan version 2.32.2)
##
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
## change `threads_per_chain` option:
## rstan_options(threads_per_chain = 1)
##
## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file
##
## Attaching package: 'rstan'
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
library(bayesplot)
```

```
## Warning: package 'bayesplot' was built under R version 4.3.3
```

```
## This is bayesplot version 1.11.1
## - Online documentation and vignettes at mc-stan.org/bayesplot
## - bayesplot theme set to bayesplot::theme_default()
##    * Does _not_ affect other ggplot2 plots
##    * See ?bayesplot_theme_set for details on theme setting
```

```r
library(coda)
```

```
## Warning: package 'coda' was built under R version 4.3.3
```

```
##
## Attaching package: 'coda'
##
## The following object is masked from 'package:rstan':
##
##     traceplot
```

Data Cleaning & Encoding

```
data <- read.csv("C:\\Users\\22840\\Desktop\\umich winter 2024\\STATS 451\\STATS451_final_projec
t\\data\\Fuel_mi_csv.csv")
filtered_data <- data %>% filter(Methane.emissions != 0)
```

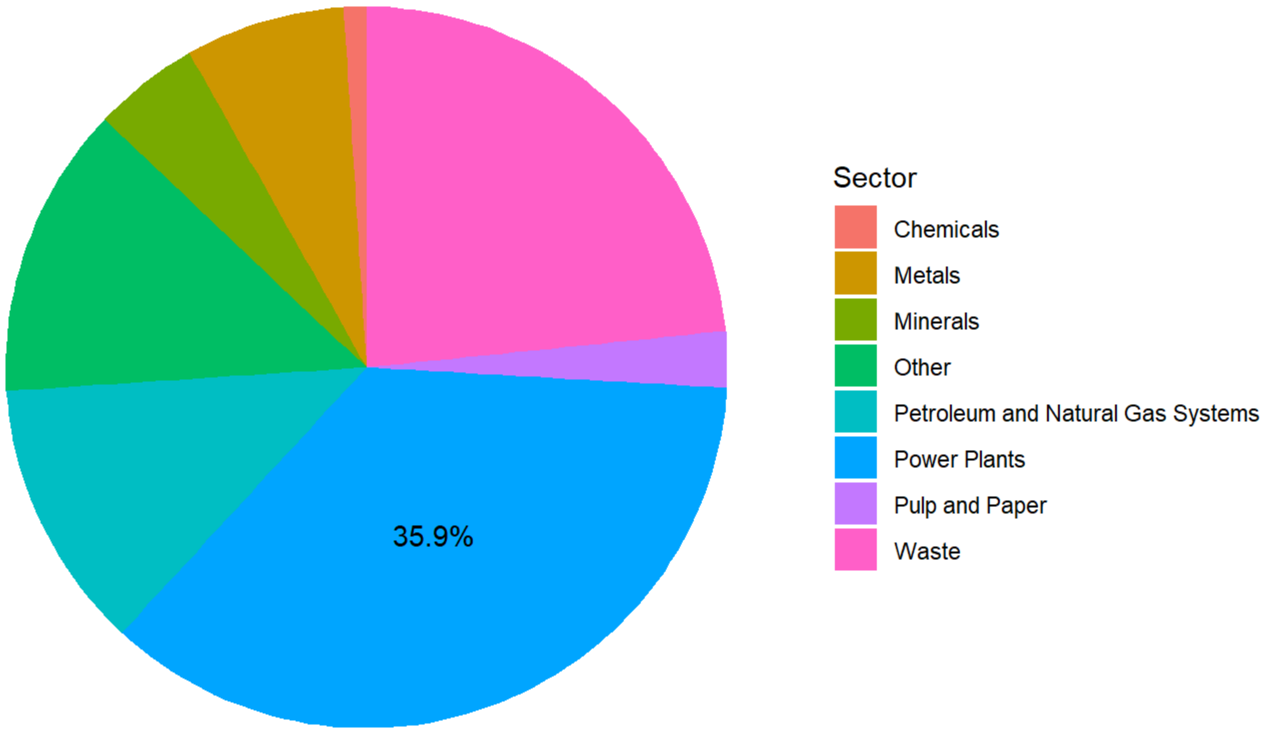
```
selected_data = filtered_data %>% dplyr::select(Year,Sector, Fuel.Type,Methane.emissions, Indust
ry.Type)
combined = data %>% filter(Sector %in% c("Chemicals","Metals", "Minerals"
                                                       ,"Other", "Petroleum and Natural Gas Syste
ms","Power Plants"
                                                       ,"Pulp and Paper","Waste"))
combined %>% head()
```

```
##   Facility.Id  FRS.Id               Facility.Name    City State
## 1     1001106 1.1e+11 48th Street Peaking Station Holland    MI
## 2     1001106 1.1e+11 48th Street Peaking Station Holland    MI
## 3     1001106 1.1e+11 48th Street Peaking Station Holland    MI
## 4     1001106 1.1e+11 48th Street Peaking Station Holland    MI
## 5     1001106 1.1e+11 48th Street Peaking Station Holland    MI
## 6     1001106 1.1e+11 48th Street Peaking Station Holland    MI
##   Primary.NAICS.Code Year Industry.Type       Sector          Unit.Name
## 1             221112 2022           C,D Power Plants                **7
## 2             221112 2022           C,D Power Plants                **7
## 3             221112 2022           C,D Power Plants                **8
## 4             221112 2022           C,D Power Plants                **8
## 5             221112 2022           C,D Power Plants                  9
## 6             221112 2022           C,D Power Plants 9 - Process Heater
##            Fuel.Type                  Specific.Fuel.Type Other.Fuel.Name
## 1        Natural Gas Natural Gas (Weighted U.S. Average)
## 2 Petroleum Products           Distillate Fuel Oil No. 2
## 3        Natural Gas Natural Gas (Weighted U.S. Average)
## 4 Petroleum Products           Distillate Fuel Oil No. 2
## 5        Natural Gas Natural Gas (Weighted U.S. Average)
## 6        Natural Gas Natural Gas (Weighted U.S. Average)
##   Blend.Fuel.Name Methane.emissions Nitrous.Oxide.emissions
## 1                              1.25                    2.98
## 2                              0.00                    0.00
## 3                              3.00                    2.98
## 4                              0.00                    0.00
## 5                             21.25                   26.82
## 6                              0.00                    0.00
```

```
sector_count <- combined %>%
  count(Sector) %>%
  mutate(Percentage = n / sum(n) * 100)

sector_count
```

```
##                               Sector    n Percentage
## 1                          Chemicals  100   1.011531
## 2                             Metals  708   7.161643
## 3                           Minerals  471   4.764313
## 4                              Other 1296  13.109448
## 5 Petroleum and Natural Gas Systems 1197  12.108032
## 6                       Power Plants 3553  35.939713
## 7                     Pulp and Paper  249   2.518713
## 8                              Waste 2312  23.386607
```

```
# find largest percentage
max_percentage <- max(sector_count$Percentage)
max_percentage <- max(sector_count$Percentage)

# pie chart
ggplot(sector_count, aes(x = "", y = n, fill = Sector)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(fill = "Sector", y = "Count", x = "") +
  theme_void() +
  geom_text(aes(label = ifelse(Percentage == max_percentage, paste0(round(Percentage, 1), "%"), 
"")),
            position = position_stack(vjust = 0.5))
```



```
print(ggplot)
```

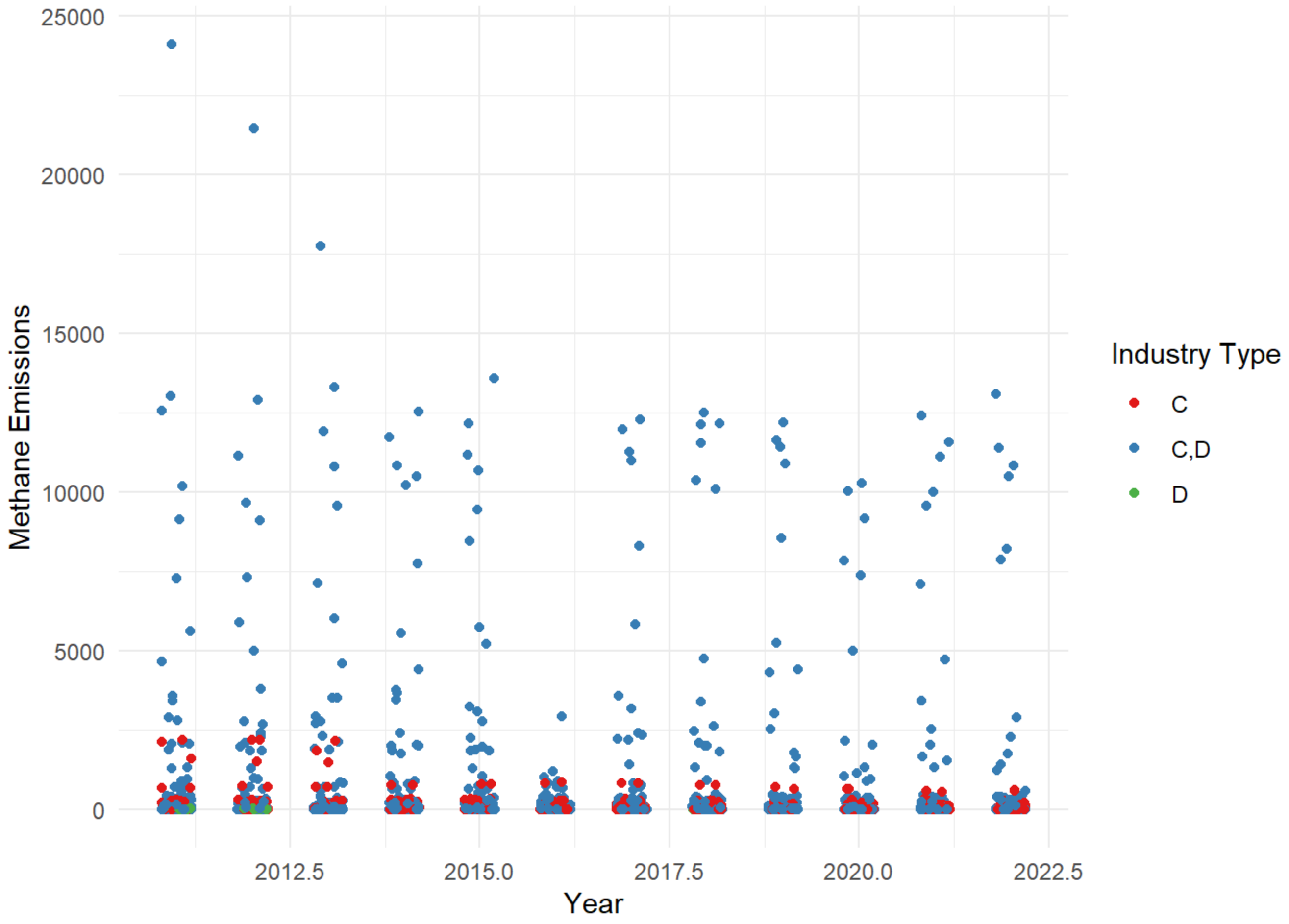
```
## function (data = NULL, mapping = aes(), ..., environment = parent.frame())
## {
##     UseMethod("ggplot")
## }
## <bytecode: 0x000001bdb95bfc90>
## <environment: namespace:ggplot2>
```

```
selected_data = filtered_data %>% dplyr::select(Year,Sector, Fuel.Type,Methane.emissions, Industry.Type)
combined_data = selected_data %>% filter(Sector %in% c("Power Plants"))
df = combined_data %>% select(Year, Fuel.Type,Methane.emissions, Industry.Type)
df %>% head()
```

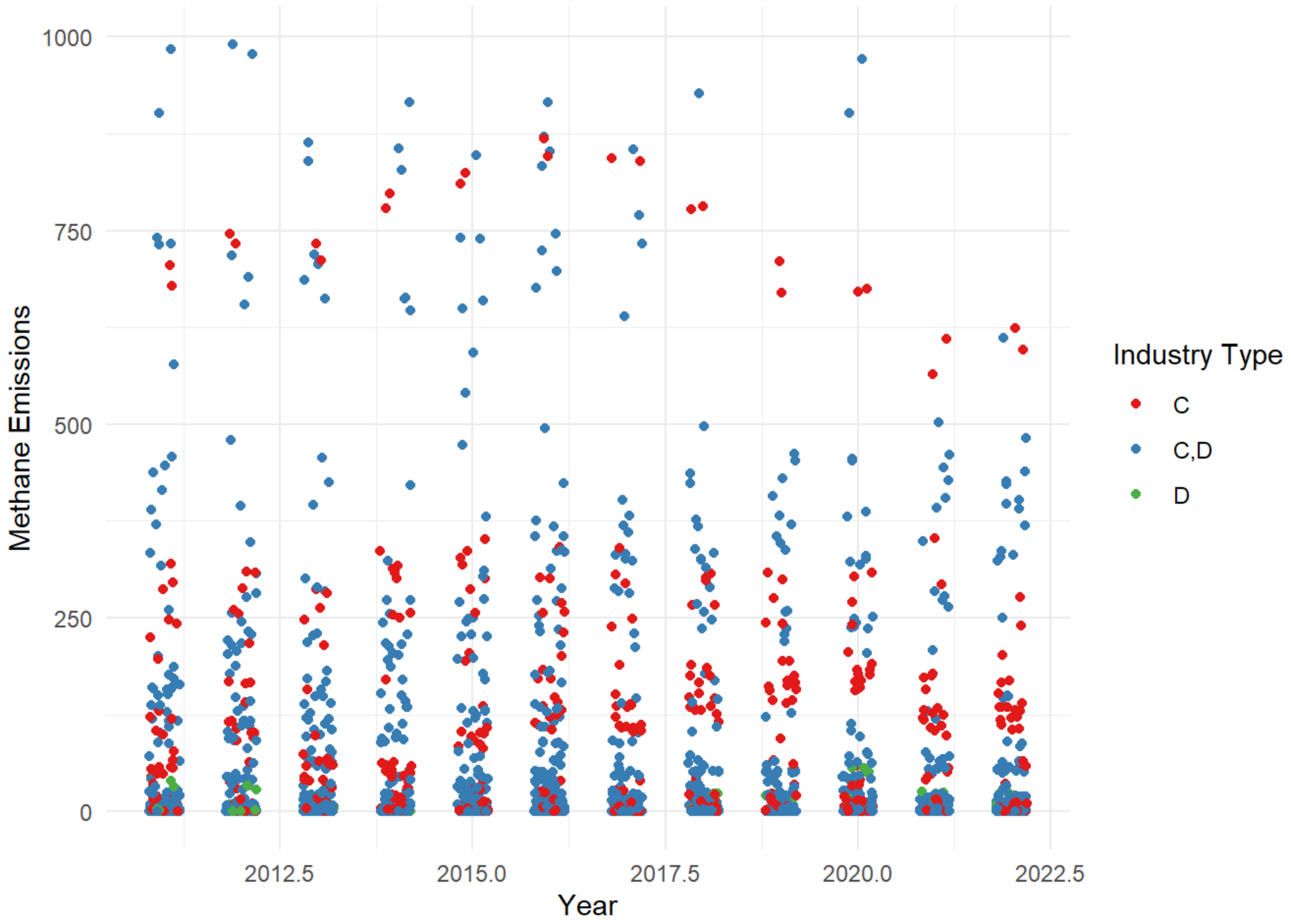
```
##   Year   Fuel.Type Methane.emissions Industry.Type
## 1 2022 Natural Gas              1.25           C,D
## 2 2022 Natural Gas              3.00           C,D
## 3 2022 Natural Gas             21.25           C,D
## 4 2021 Natural Gas              1.00           C,D
## 5 2021 Natural Gas              1.00           C,D
## 6 2021 Natural Gas             15.75           C,D
```

Data Visualization

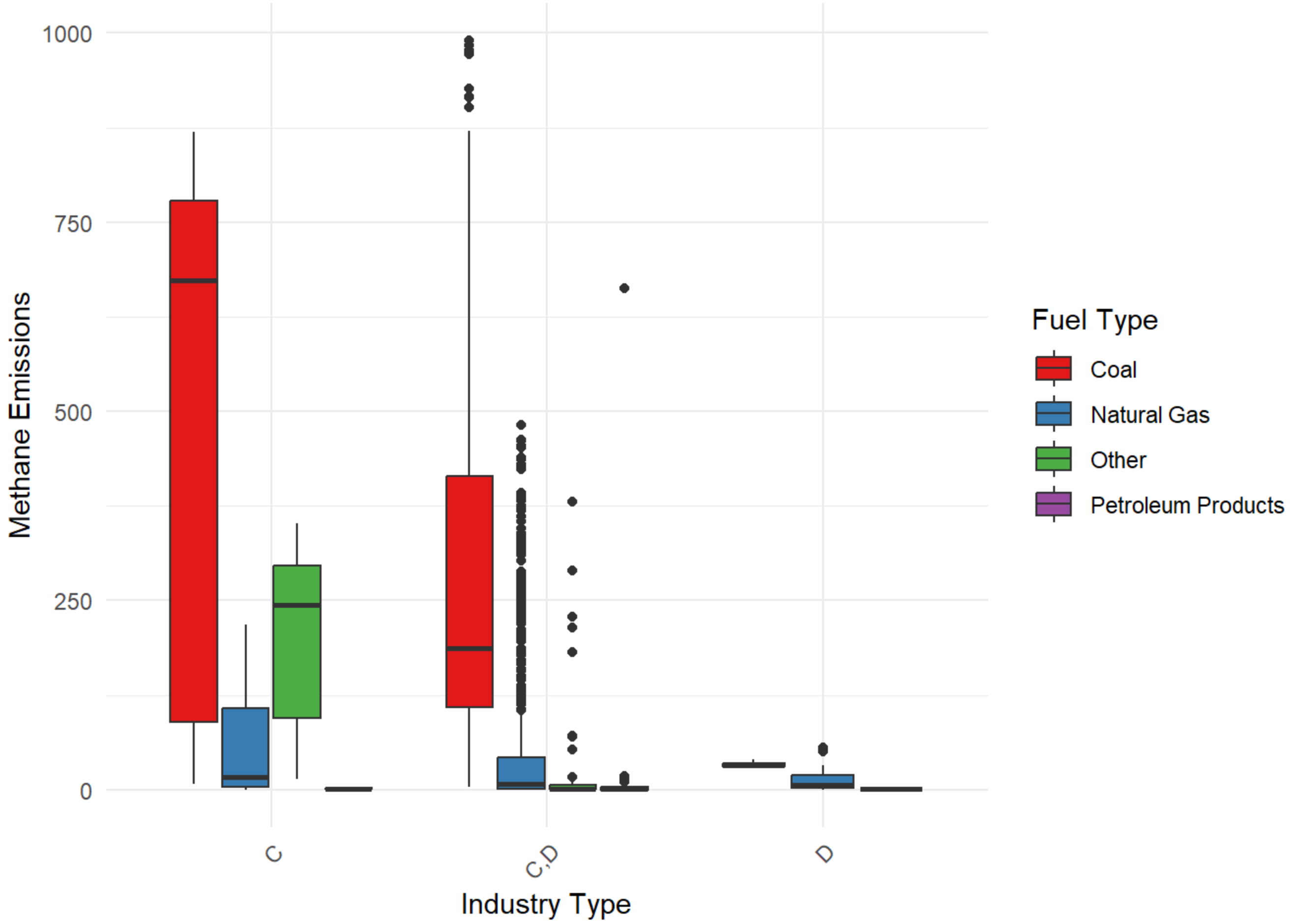
```
ggplot(df, aes(x=Year, y=Methane.emissions, color=Industry.Type)) +
  geom_jitter(width=0.2) + # Use jitter to avoid overplotting if you have discrete x values
  labs(x = "Year", y = "Methane Emissions", color = "Industry Type") +
  theme_minimal() +
  scale_color_brewer(palette = "Set1")
```



```
# Calculate the IQR for Methane Emissions
Q1 <- quantile(df$Methane.emissions, 0.25)
Q3 <- quantile(df$Methane.emissions, 0.75)
IQR <- Q3 - Q1

# Define the bounds for what constitutes an outlier
lower_bound <- Q1 - 2 * IQR
upper_bound <- 1000

# Remove outliers from the dataset
df_filtered <- df %>%
  filter(Methane.emissions >= lower_bound & Methane.emissions <= upper_bound)

# Create the plot without outliers
ggplot(df_filtered, aes(x=Year, y=Methane.emissions, color=Industry.Type)) +
  geom_jitter(width=0.2) +
  labs(x = "Year", y = "Methane Emissions", color = "Industry Type") +
  theme_minimal() +
  scale_color_brewer(palette = "Set1")
```



```
ggplot(df_filtered, aes(x=Industry.Type, y=Methane.emissions, fill=as.factor(Fuel.Type))) +
  geom_boxplot() +
  labs(x = "Industry Type", y = "Methane Emissions", fill = "Fuel Type") +
  theme_minimal() +
  scale_fill_brewer(palette = "Set1") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



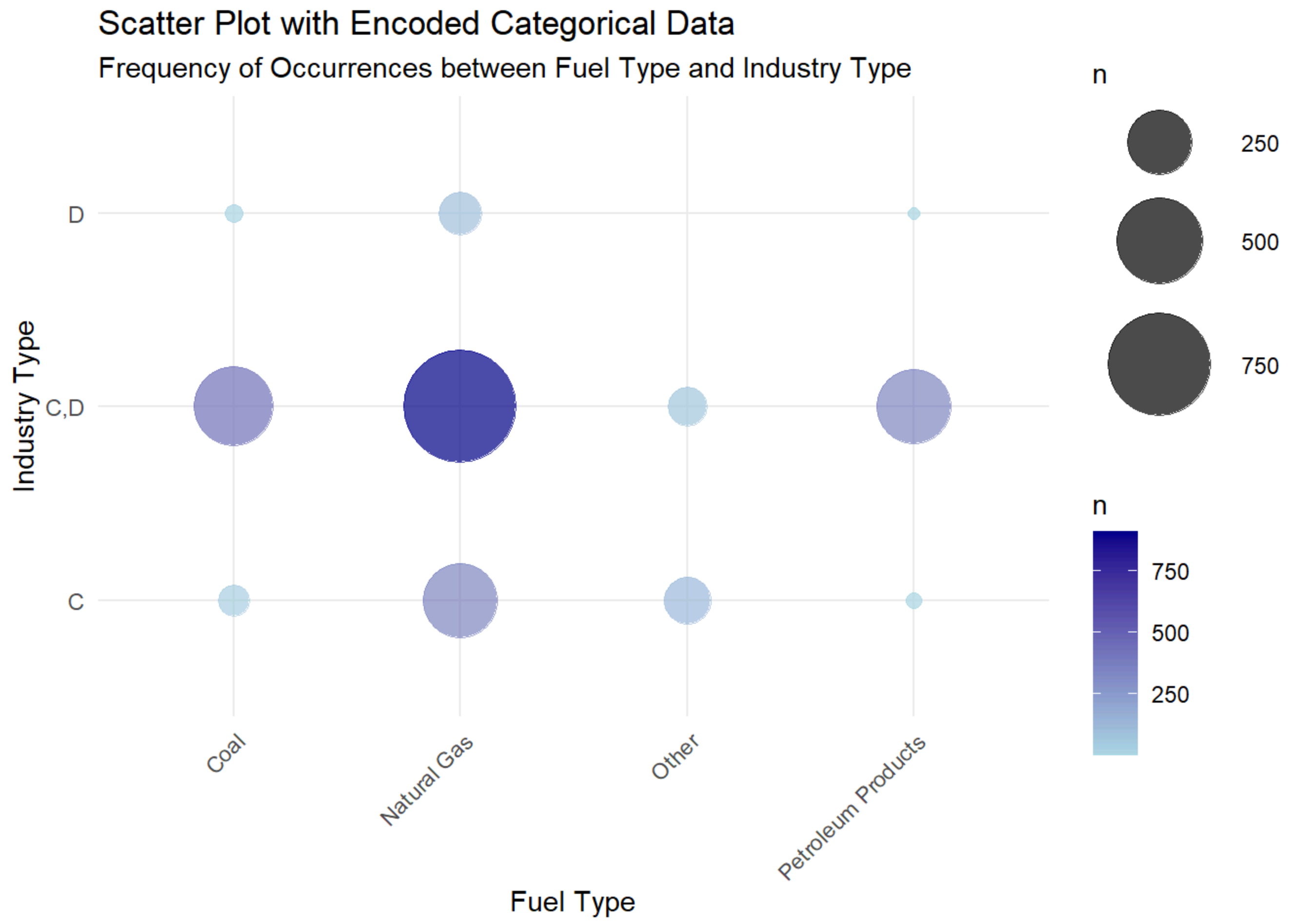
```
print(ggplot)
```

```
## function (data = NULL, mapping = aes(), ..., environment = parent.frame())
## {
##     UseMethod("ggplot")
## }
## <bytecode: 0x000001bdb95bfc90>
## <environment: namespace:ggplot2>
```

```
# Generate the counts table
counts_table <- df %>%
  count(Industry.Type, Fuel.Type)
counts_table
```

```
##    Industry.Type          Fuel.Type   n
## 1              C               Coal  36
## 2              C        Natural Gas 348
## 3              C              Other 114
## 4              C Petroleum Products   2
## 5            C,D               Coal 416
## 6            C,D        Natural Gas 911
## 7            C,D              Other  67
## 8            C,D Petroleum Products 349
## 9              D               Coal   4
## 10             D        Natural Gas  89
## 11             D Petroleum Products   1
```

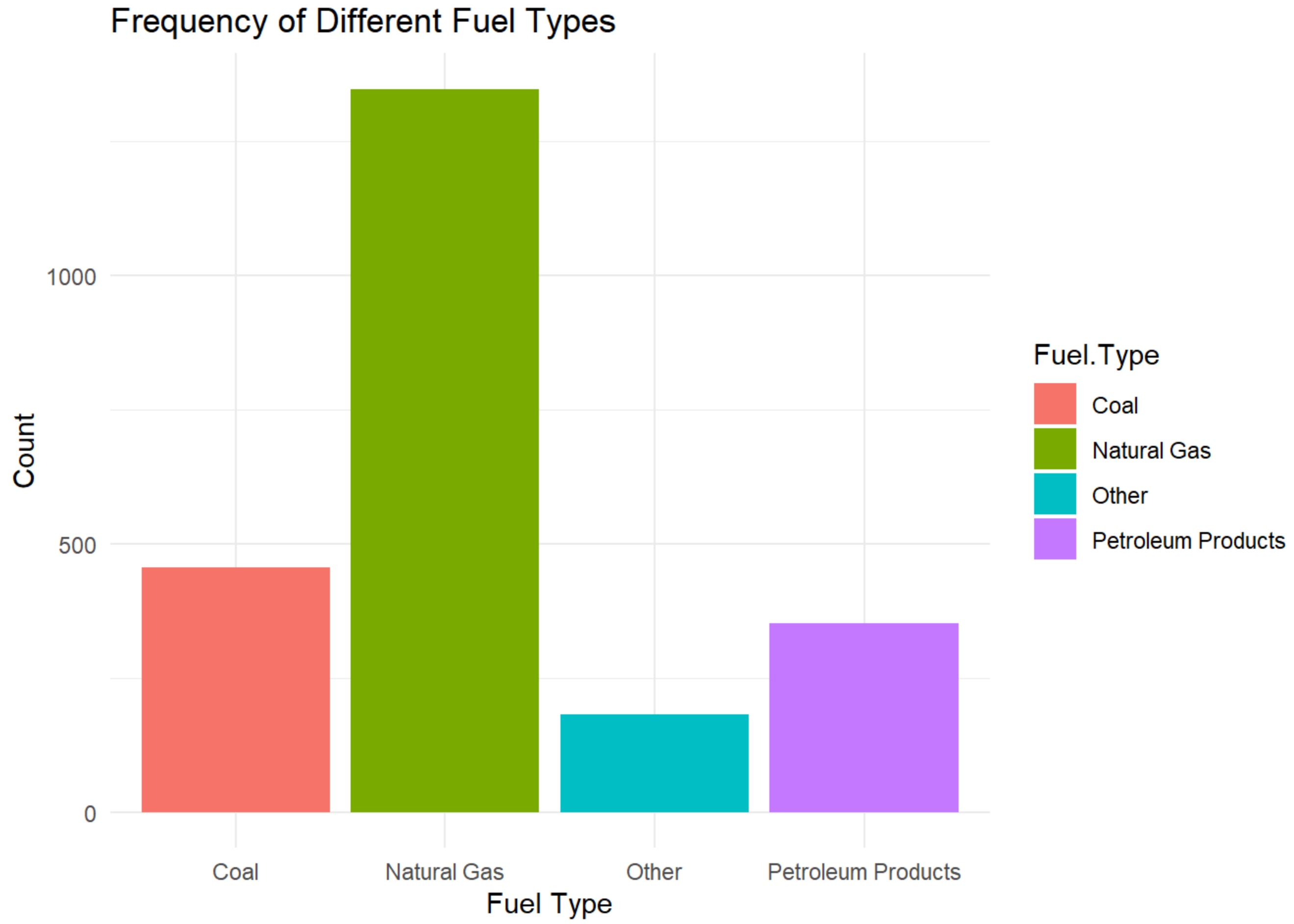
```
# Generate the scatter plot to show frequency of occurrences
ggplot(counts_table, aes(x = Fuel.Type, y = Industry.Type, size = n, color = n)) +
  geom_point(alpha = 0.7) +
  scale_color_gradient(low = "lightblue", high = "darkblue") +
  scale_size_continuous(range = c(2, 20)) +
  labs(title = 'Scatter Plot with Encoded Categorical Data',
       subtitle = 'Frequency of Occurrences between Fuel Type and Industry Type',
       x = 'Fuel Type',
       y = 'Industry Type') +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = 'right')
```



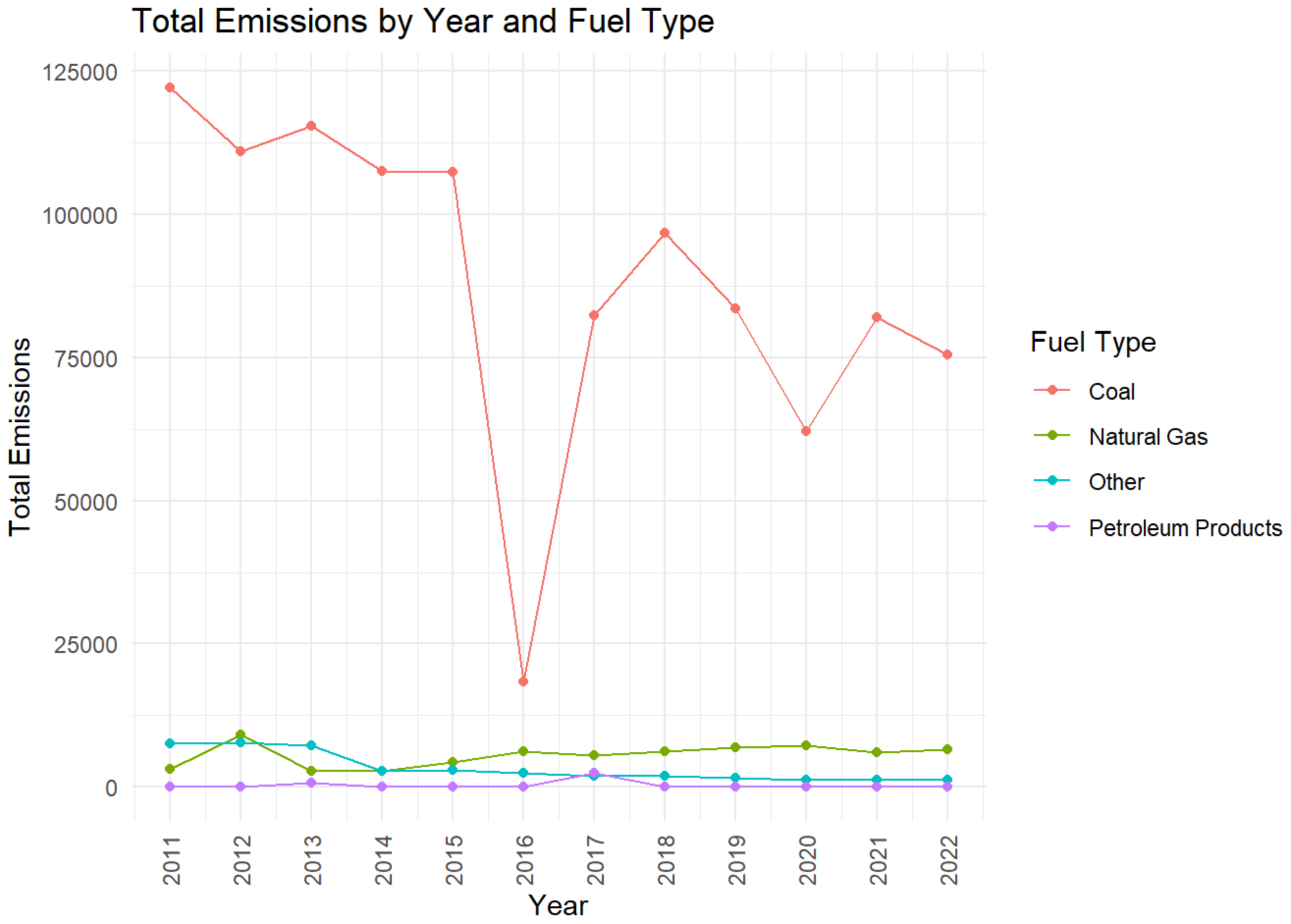
Scatter Plot with Encoded Categorical Data

Frequency of Occurrences between Fuel Type and Industry Type

```
ggplot(df, aes(x = Fuel.Type, fill = Fuel.Type)) +
  geom_bar() +
  labs(title = "Frequency of Different Fuel Types", x = "Fuel Type", y = "Count") +
  theme_minimal()
```



```
library(ggplot2)
library(dplyr)

# Aggregation of the total methane emissions by Year and the Fuel Type
emissions_by_year_fuel <- df %>%
  group_by(Year, Fuel.Type) %>%
  summarise(Total_Emissions = sum(Methane.emissions), .groups = 'drop')

# plot line chart
ggplot(emissions_by_year_fuel, aes(x = Year, y = Total_Emissions, group = Fuel.Type, color = Fue
l.Type)) +
  geom_line() +
  geom_point() +
  scale_x_continuous(breaks = seq(min(emissions_by_year_fuel$Year), max(emissions_by_year_fuel$Y
ear), by = 1)) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
  labs(title = 'Total Emissions by Year and Fuel Type',
       x = 'Year', y = 'Total Emissions', color = 'Fuel Type')
```


Total Emissions by Year and Fuel Type
Total Emissions
125000
100000
75000
50000
25000
0
Year
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
Fuel Type
Coal
Natural Gas
Other
Petroleum Products