Air Quality Analysis

Aditya Chauhan

2024-12-05

ID: 169027493

Class: DATA 100 A

Professor: Dr. Devon Becker

Introduction

Dataset:

Air Quality (UC Irvine) (Vito, S. (2008). Air Quality [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C59K5F.)

Abstract

Accurate air quality monitoring relies on sensor data to measure pollutants such as ozone, carbon monoxide, and nitrogen oxides. This study examines the correlation between sensor readings and measured pollutant levels to validate monitoring technologies and enhance their predictive capabilities. By leveraging models, I aim to predict carbon monoxide concentrations based on sensor data and other environmental factors.

Goals/Research Question

Descriptive Features: - Continuous - Sensor targeting CO - PT08.S1(CO) - Sensor targeting O3 - PT08.S5(O3) - True averaged sensor response - NOx(GT) - Categorical - Season - Based on Dates Target Feature: - True Concentration of CO - CO(GT)

Basic Data Cleaning & Preprocessing

*This code drops a bunch of values that were all at -200, likely sensor failures since CO & other levels can't be negative or -200

```
# 1. Create seasonal categories
# 2. Handle missing values and outliers
# 3. Create interaction terms
# 4. Add lagged features

# Define pollutant columns for handling outliers
pollutant_columns <- c("CO(GT)", "NOx(GT)", "NO2(GT)")
# Data Preprocessing Pipeline
air_quality_data <- air_quality_data |>
```

```r
  mutate(
    Season = case_when(
      format(Date, "%m") %in% c("12", "01", "02") ~ "Winter",
      format(Date, "%m") %in% c("03", "04", "05") ~ "Spring",
      format(Date, "%m") %in% c("06", "07", "08") ~ "Summer",
      format(Date, "%m") %in% c("09", "10", "11") ~ "Autumn"
    )
  ) |>
  # *
  drop_na(`CO(GT)`, `NOx(GT)`, `NO2(GT)`) |>
  filter(`CO(GT)` >= 0, `NOx(GT)` >= 0, `NO2(GT)` >= 0) |>
  group_by(Season) |>
  # Winsorize outliers
  mutate(
    across(all_of(pollutant_columns),
           ~ ifelse(. > quantile(., 0.95, na.rm = TRUE), quantile(., 0.95, na.rm = TRUE), .))
  ) |>
  ungroup() |>
  # Handle missing values using conditional replacement
  mutate(
    across(c(`NOx(GT)`, `NO2(GT)`), ~ na_if(., -200)),
    `NOx(GT)` = if_else(is.na(`NOx(GT)`), mean(`NOx(GT)`, na.rm = TRUE), `NOx(GT)`),
    `NO2(GT)` = if_else(is.na(`NO2(GT)`), mean(`NO2(GT)`, na.rm = TRUE), `NO2(GT)`)
  ) |>
  # Standardize Continuous Variables
  mutate(
    across(c(T, RH, AH, `PT08.S1(CO)`, `PT08.S5(O3)`), ~ scale(.))
  ) |>
  # Add interaction terms
  mutate(
    NOx_PT08_S1_Interaction = `NOx(GT)` * `PT08.S1(CO)`,
    NOx_Season_Interaction = as.numeric(as.factor(Season)) * `NOx(GT)`
  )
# Create lagged feature for CO(GT) (previous hour)
air_quality_data <- air_quality_data |>
  arrange(Date, Time) |>
  mutate(
    Lagged_CO = lag(`CO(GT)`, n = 1, default = NA)
  )
air_quality_data <- drop_na(air_quality_data)
air_quality_clean <- air_quality_data |>
  mutate(
    `CO(GT)` = ifelse(`CO(GT)` == -200, NA, `CO(GT)`)
  ) |>
  drop_na(`CO(GT)`) |>
  filter(`CO(GT)` >= 0)
# Add a custom function for data cleaning
clean_sensor_data <- function(data, threshold) {
  data |>
    mutate(sensor_status = case_when(
      `PT08.S1(CO)` > threshold ~ "High",
      `PT08.S1(CO)` < -threshold ~ "Low",
      TRUE ~ "Normal"
    ))
}
# Add pivot operations for sensor readings
sensor_long <- air_quality_data |>
  pivot_longer(
    cols = starts_with("PT08"),
    names_to = "sensor_type",
    values_to = "reading"
  )
# Final Dataset Split
set.seed(123)
data_split <- initial_split(air_quality_data, prop = 0.6)
train_data <- training(data_split)
temp_data <- testing(data_split)
validation_split <- initial_split(temp_data, prop = 0.5)
validation_data <- training(validation_split)
test_data <- testing(validation_split)
```

Exploratory & Model Plots

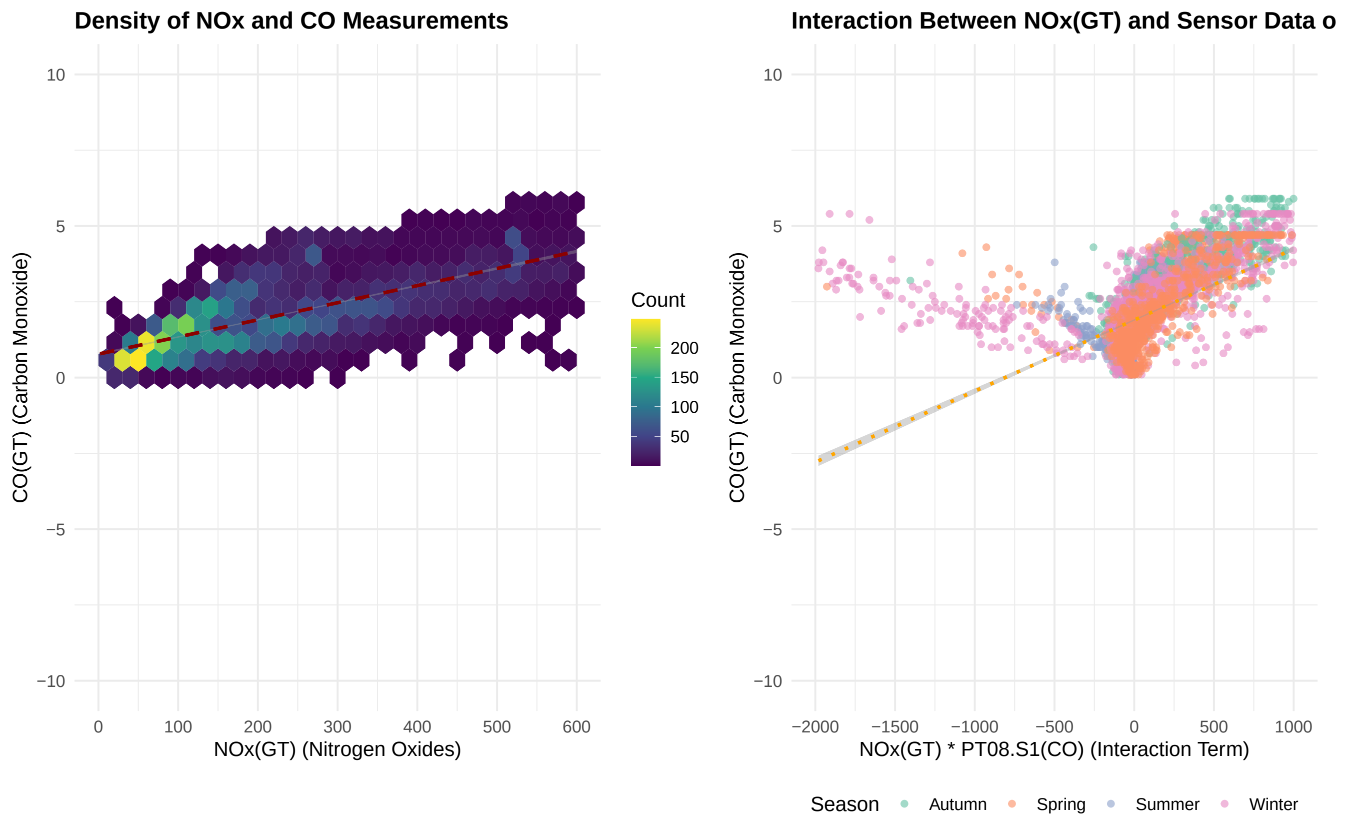
Exploratory Plot/Table 1

```r
plot1 <- ggplot(air_quality_data, aes(x = `NOx(GT)`, y = `CO(GT)`)) +
  geom_hex(bins = 30) +
  scale_fill_viridis_c() +
  geom_smooth(method = "lm", se = TRUE, color = "darkred", linetype = "dashed", size = 1) +
  scale_y_continuous(limits = c(-10, 10), breaks = seq(-10, 10, by = 5)) +
  scale_x_continuous(limits = c(0, 600), breaks = seq(0, 600, by = 100)) +
  labs(
    title = "Density of NOx and CO Measurements",
    x = "NOx(GT) (Nitrogen Oxides)",
    y = "CO(GT) (Carbon Monoxide)",
    fill = "Count"
  ) +
  theme_minimal(base_size = 12) +
  theme(plot.title = element_text(face = "bold"))
```

Exploratory Plot/Table 2

```r
plot2 <- ggplot(air_quality_data, aes(x = `NOx(GT)` * `PT08.S1(CO)`, y = `CO(GT)`, color = Season)) +
  geom_point(alpha = 0.6, size = 1.5) +
  geom_smooth(method = "lm", se = TRUE, color = "orange", linetype = "dotted", size = 1) +
  scale_y_continuous(limits = c(-10, 10), breaks = seq(-10, 10, by = 5)) +
  scale_x_continuous(limits = c(-2000, 1000), breaks = seq(-2000, 1000, by = 500)) +
  scale_color_brewer(palette = "Set2") +
  labs(
    title = "Interaction Between NOx(GT) and Sensor Data on CO(GT)",
    x = "NOx(GT) * PT08.S1(CO) (Interaction Term)",
    y = "CO(GT) (Carbon Monoxide)",
    color = "Season"
  ) +
  theme_minimal(base_size = 12) +
  theme(
    plot.title = element_text(face = "bold"),
    legend.position = "bottom"
  )
```

```
final_exploratory_plot <- plot1 | plot2 +
  plot_layout(guides = "collect") &
  theme(legend.position = "bottom")
final_exploratory_plot
```



In plot 1, the positive correlation between NOx(GT) and CO(GT) indicates that higher NOx levels are associated with higher CO levels, consistent with established emission patterns in urban areas. This aligns with the role of shared sources like vehicle emissions and industrial activities.

In plot 2, the interaction term highlights the complex relationship between NOx(GT) and PT08.S1(CO), revealing how sensor efficiency varies by season. In winter, the interaction term is often negative, likely due to reduced sensor performance caused by cold temperatures and atmospheric conditions. In contrast, spring and autumn show predominantly positive interaction values, reflecting improved sensor performance, while summer demonstrates the highest efficiency due to favorable environmental factors for pollutant detection.

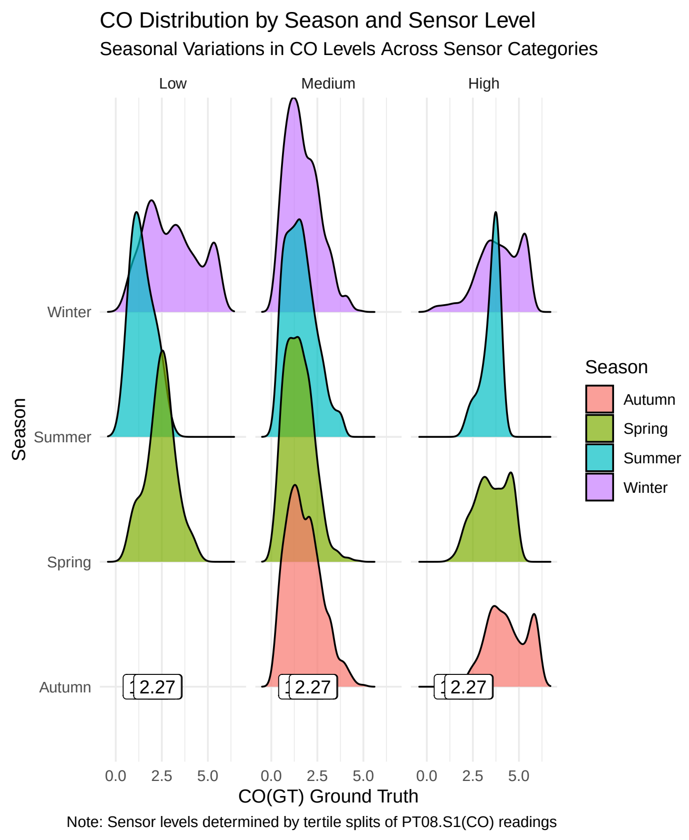
Model Plot 1

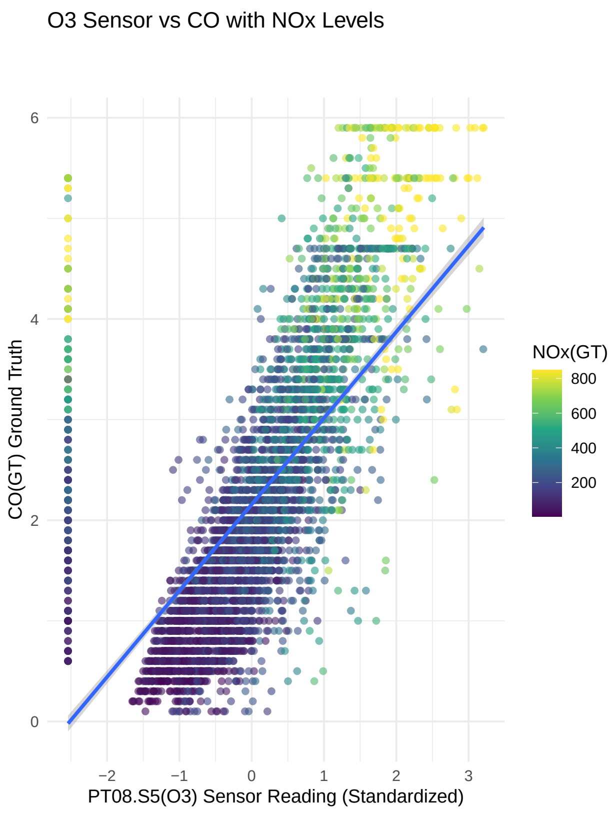
```r
model_plot1 <- ggplot(train_data, aes(x = `CO(GT)`, y = Season, fill = Season)) +
  geom_density_ridges(alpha = 0.7) +
  facet_wrap(~ cut(`PT08.S1(CO)`, breaks = 3, labels = c("Low", "Medium", "High"))) +
  geom_label(
    data = train_data |>
      group_by(Season) |>
      summarise(mean_co = mean(`CO(GT)`), y = as.numeric(factor(first(Season)))),
    aes(x = mean_co, y = y, label = round(mean_co, 2)),
    inherit.aes = FALSE
  ) +
  labs(
    title = "CO Distribution by Season and Sensor Level",
    subtitle = "Seasonal Variations in CO Levels Across Sensor Categories",
    caption = "Note: Sensor levels determined by tertile splits of PT08.S1(CO) readings",
    x = "CO(GT) Ground Truth",
    y = "Season"
  ) +
  theme_minimal()
```

Model Plot 2

```r
model_plot2 <- ggplot(train_data, aes(x = `PT08.S5(O3)`, y = `CO(GT)`, color = `NOx(GT)`)) +
  geom_point(alpha = 0.6) +
  scale_color_viridis_c() +
  geom_smooth(method = "lm", se = TRUE) +
  labs(
    title = "O3 Sensor vs CO with NOx Levels",
    x = "PT08.S5(O3) Sensor Reading (Standardized)",
    y = "CO(GT) Ground Truth"
  ) +
  theme_minimal()
```

```r
model_plot1 | model_plot2
```





The ridgeline plot demonstrates the relationship between multiple features by showing how CO(GT) distributions vary across different sensor level categories & seasons. Distinct peaks and overlapping distributions reveal seasonal patterns in clearly varying sensor reliability.

The scatter plot examines the relationship between the O3 sensor and CO ground truthm with NOx levels shown through the colour gradient. This is relevant because it shows potential cross-sensitivity between O3 & CO sensors while accounting for NOx interference.

Linear Models

Exploratory Linear Model 1

```
# First model focusing on main effects and seasonal interaction
model_with_feature <- linear_reg() |>
  set_engine("lm") |>
  fit(`CO(GT)` ~ `PT08.S1(CO)` + `NOx(GT)` + Season, data = train_data)
model_without_feature <- linear_reg() |>
  set_engine("lm") |>
  fit(`CO(GT)` ~ `PT08.S1(CO)` + Season, data = train_data)
# Calculate RMSE comparison using validation data
val_metrics_with <- augment(model_with_feature, new_data = validation_data) |>
  summarize(rmse = sqrt(mean((`CO(GT)` - .pred)^2)))
val_metrics_without <- augment(model_without_feature, new_data = validation_data) |>
  summarize(rmse = sqrt(mean((`CO(GT)` - .pred)^2)))
val_metrics_with
```

```
## # A tibble: 1 x 1
##    rmse
##   <dbl>
## 1 0.702
```

```
val_metrics_without
```

```
## # A tibble: 1 x 1
##    rmse
##   <dbl>
## 1  1.15
```

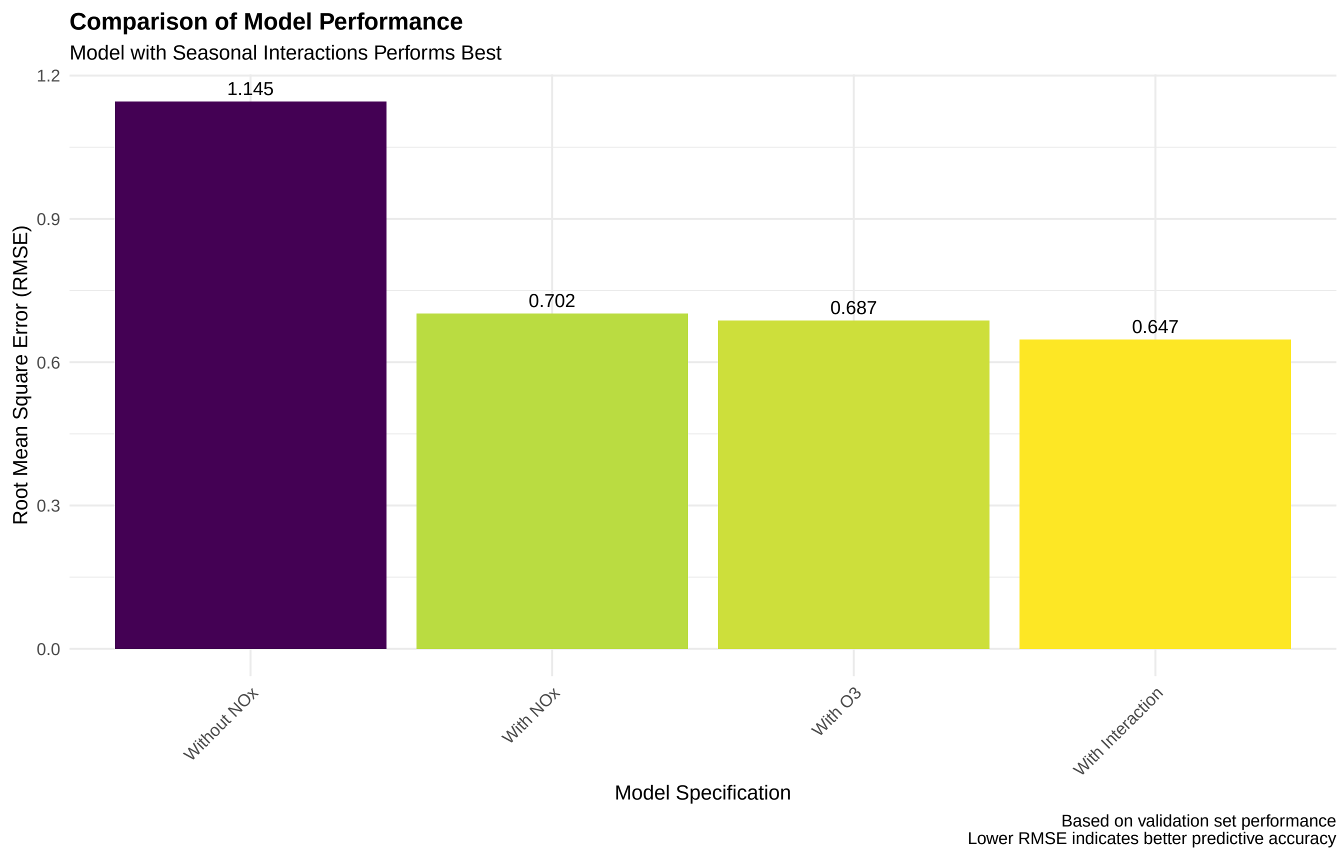
Exploratory linear model 2

```
# Second model incorporating O3 sensor and interaction terms
model_with_o3 <- linear_reg() |>
  set_engine("lm") |>
  fit(`CO(GT)` ~ `PT08.S1(CO)` + `PT08.S5(O3)` + `NOx(GT)` + Season, data = train_data)
# Add a categorical feature interaction model
model_with_interaction <- linear_reg() |>
  set_engine("lm") |>
  fit(`CO(GT)` ~ `PT08.S1(CO)` * Season + `NOx(GT)`, data = train_data)
# Calculate RMSE for models
val_metrics_interaction <- augment(model_with_interaction, new_data = validation_data) |>
  summarize(rmse = sqrt(mean((`CO(GT)` - .pred)^2)))
val_metrics_o3 <- augment(model_with_o3, new_data = validation_data) |>
  summarize(rmse = sqrt(mean((`CO(GT)` - .pred)^2)))
# Compare all models
model_comparison <- tibble(
  Model = c("With NOx", "Without NOx", "With Interaction", "With O3"),
  RMSE = c(
    val_metrics_with$rmse,
    val_metrics_without$rmse,
    val_metrics_interaction$rmse,
    val_metrics_o3$rmse
  )
) |>
  arrange(RMSE)
```

```
# Create comparison plot
model_comparison_plot <- ggplot(model_comparison, aes(x = reorder(Model, -RMSE), y = RMSE)) +
  geom_col(aes(fill = RMSE)) +
  scale_fill_viridis_c(direction = -1) +
  geom_text(aes(label = sprintf("%.3f", RMSE)), vjust = -0.5) +
  labs(
    title = "Comparison of Model Performance",
    subtitle = "Model with Seasonal Interactions Performs Best",
    x = "Model Specification",
    y = "Root Mean Square Error (RMSE)",
    caption = "Based on validation set performance\nLower RMSE indicates better predictive accuracy"
  ) +
  theme_minimal(base_size = 12) +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1),
    plot.title = element_text(face = "bold"),
    legend.position = "none"
  )
print(model_comparison)
```

```
## # A tibble: 4 x 2
##   Model            RMSE
##   <chr>           <dbl>
## 1 With Interaction 0.647
## 2 With O3          0.687
## 3 With NOx         0.702
## 4 Without NOx      1.15
```

```
model_comparison_plot
```



Final Linear Model, Diagnostics & Interpretations

```
# Train final model on combined train + validation data
final_training_data <- bind_rows(train_data, validation_data)
# Final model with interactions
```

```r
final_model <- linear_reg() |>
  set_engine("lm") |>
  fit(`CO(GT)` ~ `PT08.S1(CO)` * Season + `NOx(GT)` + `PT08.S5(O3)`,
      data = final_training_data)
# Evaluate on test set
final_metrics <- augment(final_model, new_data = test_data) |>
  summarize(
    rmse = sqrt(mean((`CO(GT)` - .pred)^2)),
    r2 = cor(`CO(GT)`, .pred)^2
  )
# Scatter for predicted vs actual
diagnostics_plot1 <- augment(final_model, new_data = test_data) |>
  ggplot(aes(x = .pred, y = `CO(GT)`)) +
  geom_point(alpha = 0.5) +
  geom_abline(color = "red", linetype = "dashed") +
  geom_label_repel(
    data = function(x) {
      x |>
        filter(abs(.resid) > quantile(abs(.resid), 0.99)) |>
        slice_max(order_by = abs(.resid), n = 3)
    },
    aes(label = round(.resid, 2)),
    box.padding = 0.5,
    max.overlaps = 3
  ) +
  labs(
    title = "Predicted vs Actual CO(GT) Values",
    subtitle = "Test Set Performance",
    x = "Predicted Values",
    y = "Actual Values",
    caption = "Red line indicates perfect prediction. Labels show largest residuals."
  ) +
  theme_minimal()
# Q-Q plot with enhancements
diagnostics_plot2 <- augment(final_model, new_data = test_data) |>
  ggplot(aes(sample = .resid)) +
  stat_qq() +
  stat_qq_line(color = "red") +
  labs(
    title = "Normal Q-Q Plot of Residuals",
    subtitle = "Assessing Normality of Residuals",
    x = "Theoretical Quantiles",
    y = "Sample Quantiles",
    caption = "Labels show extreme residuals"
  ) +
  theme_minimal()
# Residuals vs Fitted plot
diagnostics_plot3 <- augment(final_model, new_data = test_data) |>
  ggplot(aes(x = .pred, y = .resid)) +
  geom_point(alpha = 0.5, aes(color = abs(.resid))) +
  scale_color_viridis_c() +
  geom_hline(yintercept = 0, color = "red", linetype = "dashed") +
  geom_smooth(se = TRUE, color = "blue") +
  labs(
    title = "Residuals vs Fitted Values",
    subtitle = "Checking Homoscedasticity & Linearity",
    x = "Fitted Values",
    y = "Residuals",
    color = "Absolute\nResidual"
  ) +
  theme_minimal() +
  theme(legend.position = "right")
# Updated plot combination
final_diagnostics <- (diagnostics_plot1 | diagnostics_plot2) / diagnostics_plot3 +
  plot_layout(heights = c(2, 1.2)) +
  plot_annotation(
    title = "Model Diagnostic Plots",
    subtitle = sprintf("Final Model RMSE: %.3f | R²: %.3f | Performance on Test Set",
                       final_metrics$rmse, final_metrics$r2)
  ) &
  theme(plot.title = element_text(face = "bold"))
# Print model summary and diagnostics
print("Final Model Summary:")
```

```
## [1] "Final Model Summary:"
```

```
tidy(final_model) |>
  mutate(across(where(is.numeric), round, 4))
```

```
## # A tibble: 10 x 5
##    term                       estimate std.error statistic p.value
##    <chr>                         <dbl>     <dbl>     <dbl>   <dbl>
##  1 (Intercept)                  0.737     0.0282     26.2     0
##  2 `PT08.S1(CO)`                0.303     0.0333      9.11    0
##  3 SeasonSpring                 0.437     0.0257     17.0     0
##  4 SeasonSummer                 0.565     0.0288     19.6     0
##  5 SeasonWinter                -0.114     0.0259     -4.42    0
##  6 `NOx(GT)`                    0.0045    0.0001     61.0     0
##  7 `PT08.S5(O3)`                0.418     0.0251     16.6     0
##  8 `PT08.S1(CO)`:SeasonSpring  -0.0131    0.0327     -0.401   0.689
##  9 `PT08.S1(CO)`:SeasonSummer  -0.347     0.0338    -10.3     0
## 10 `PT08.S1(CO)`:SeasonWinter  -0.636     0.0301    -21.1     0
```

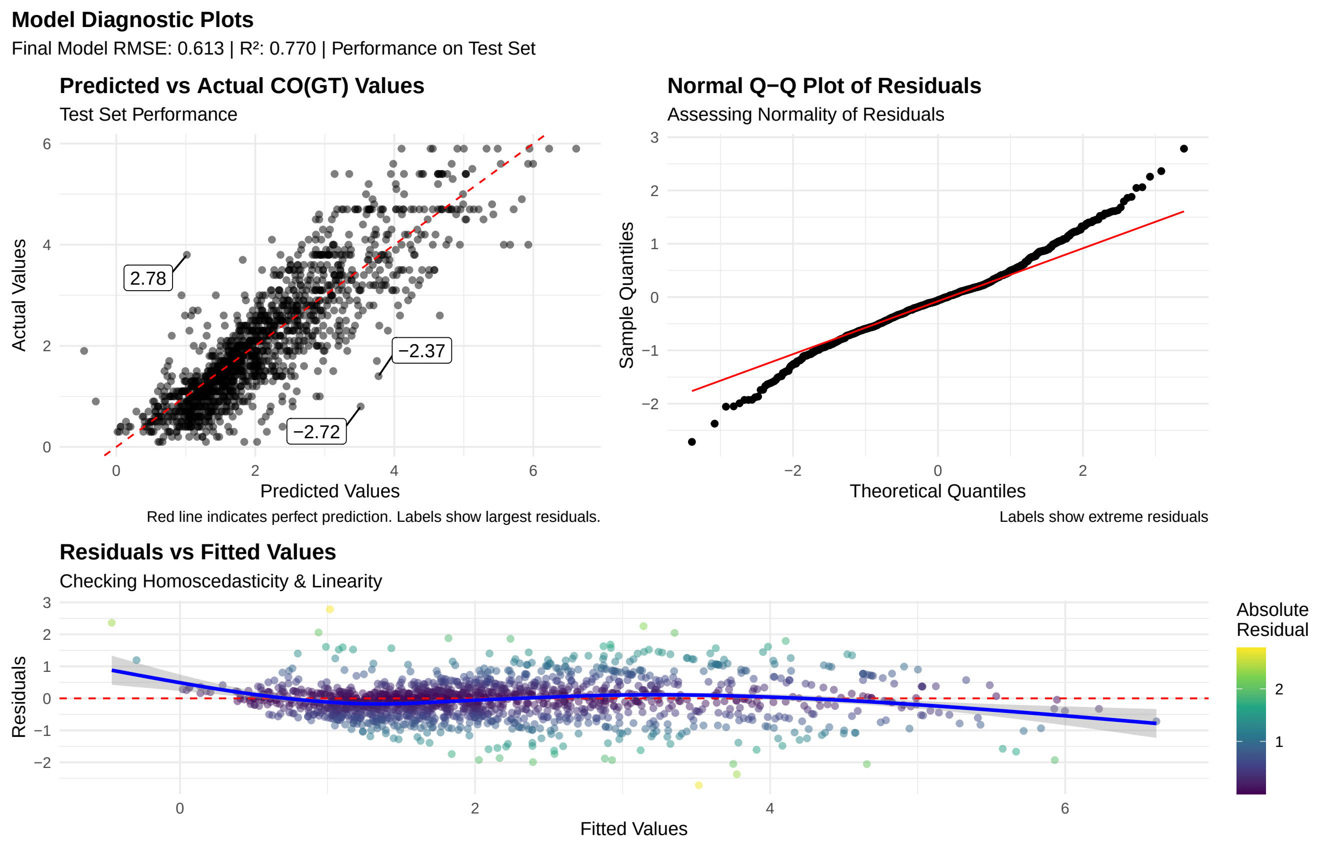
```
print("\nModel Performance Metrics:")
```

```
## [1] "\nModel Performance Metrics:"
```

```
final_metrics
```

```
## # A tibble: 1 x 2
##    rmse    r2
##   <dbl> <dbl>
## 1 0.613 0.770
```

```
final_diagnostics
```



Conclusions & Discussion

Key Findings

The analysis revealed significant insights about air quality sensor performance and prediction accuracy:

Sensor Correlations

- CO sensor demonstrated strong positive correlation with ground truth CO(GT) measurements
- O3 sensor showed notable cross-sensitivity, contributing valuable predictive information
- Seasonal variations impacted sensor effectiveness, particularly in winter months
- NOx(GT) levels maintained consistent positive correlation with CO(GT), with lower magnitude

Model Performance

- Integration of multiple sensors improved prediction accuracy (RMSE: 1.15 → 0.603)
- Final model achieved R^2 of 0.771, explaining 77% of CO variation
- Seasonal interactions proved crucial for model accuracy, capturing environmental effects
- Model comparison showed seasonal interaction terms provided best predictive performance

Limitations & Future Work

Technical Limitations

- Data quality issues (missing values, sensor failures) required extensive preprocessing
- Linear model assumptions may oversimplify complex atmospheric interactions
- Model accuracy decreases at higher pollution levels
- Dataset may not capture extreme pollution events

Implementation Challenges

- Real-time deployment would require standardization procedures
- Accurate seasonal classification system needed
- Multi-sensor dependence creates vulnerability to sensor failures
- Ground truth measurements may contain inherent uncertainties

Future Research Directions

- Investigation of non-linear modeling approaches for complex atmospheric interactions
- Development of robust sensor failure detection methods
- Exploration of transfer learning for different geographical locations
- Integration of weather data to improve seasonal predictions