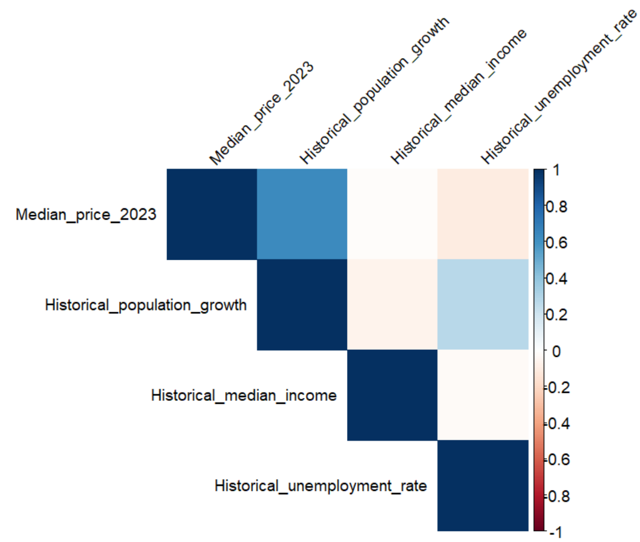
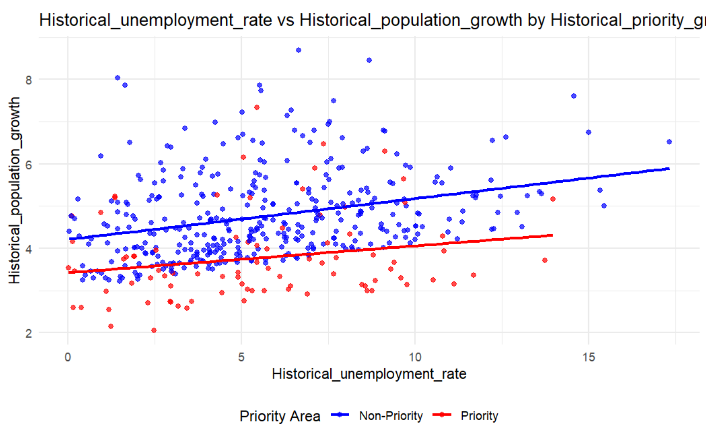
1. Data Summary

Three datasets have been provided about real estate in Victoria, Australia which have been merged on the 'Suburb_name' column. A total of one missing value and 1 extreme outlier were found which was handled using the median of their respective column. Negative values in unemployment_rate is not possible and was converted to positive since we can't reduce any datapoints.

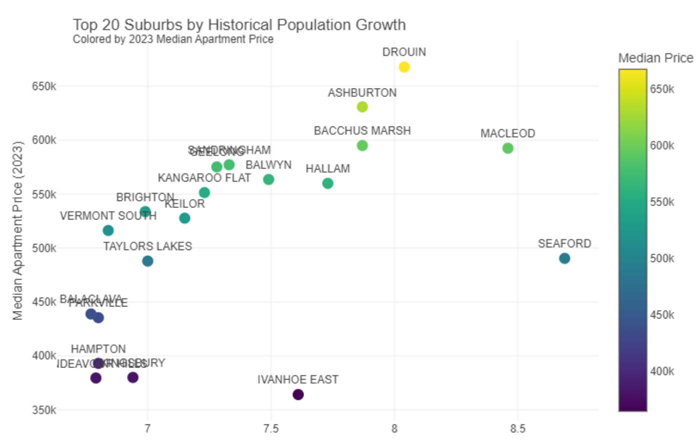
Historical_population_growth showed the highest correlation with Median_price_2023 of 0.63663424. Historical_median_income showed a negligible amount of correlation with Median_price_2023.





The nonpriority growth areas showed a strong relationship to population growth as seen in the graph, meaning they are viewed as prospects despite the high unemployment in the area.

In all of the suburbs, Seaford seems to have the highest increase in population and Drouin has the highest Apartment prices exceeding 650K. Ivanhoe East might be a top contender since it has high population growth and a low price for its apartments.



2. <u>Model Estimation</u>

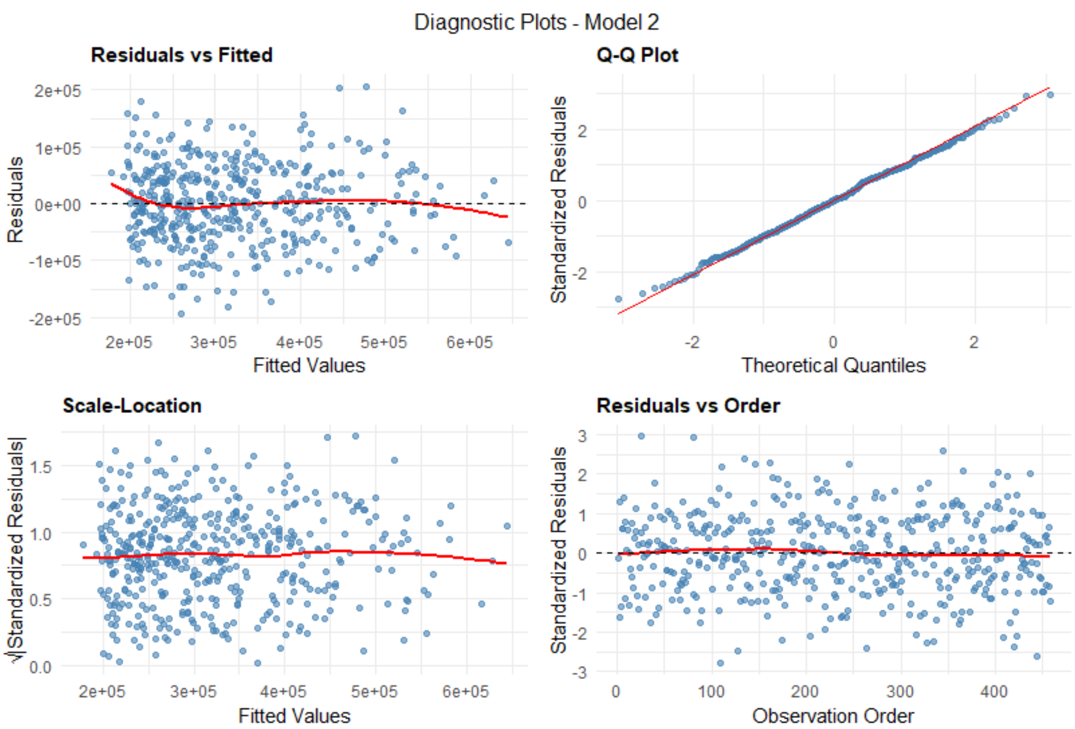
I estimated two regression models one being linear (model_1). and other being quadratic (model_2). Model_2 performed slightly better than the previous.

```
>   print(model_metrics, row.names = FALSE, digits = 4)
                 Model     R2 Adj_R2   AIC   BIC  RMSE   RSE
    Model 1 (Linear) 0.6421 0.6389 11535 11560 70333 70720
 Model 2 (Quadratic) 0.6521 0.6475 11526 11559 69339 69875
```

$$
\begin{aligned}
\text{Predicted Median Price 2023} = \beta_0 \; + \; & \beta_1 \cdot \text{Historical Population Growth} \\
+ \; & \beta_2 \cdot (\text{Historical Population Growth})^2 \\
+ \; & \beta_3 \cdot \text{Historical Unemployment Rate} \\
+ \; & \beta_4 \cdot \text{Historical Priority Growth Area} \\
+ \; & \beta_5 \cdot (\text{Historical Unemployment Rate} \times \text{Historical Population Growth}) \\
+ \; & \beta_6 \cdot (\text{Historical Population Growth} \times \text{Historical Priority Growth Area})
\end{aligned}
$$

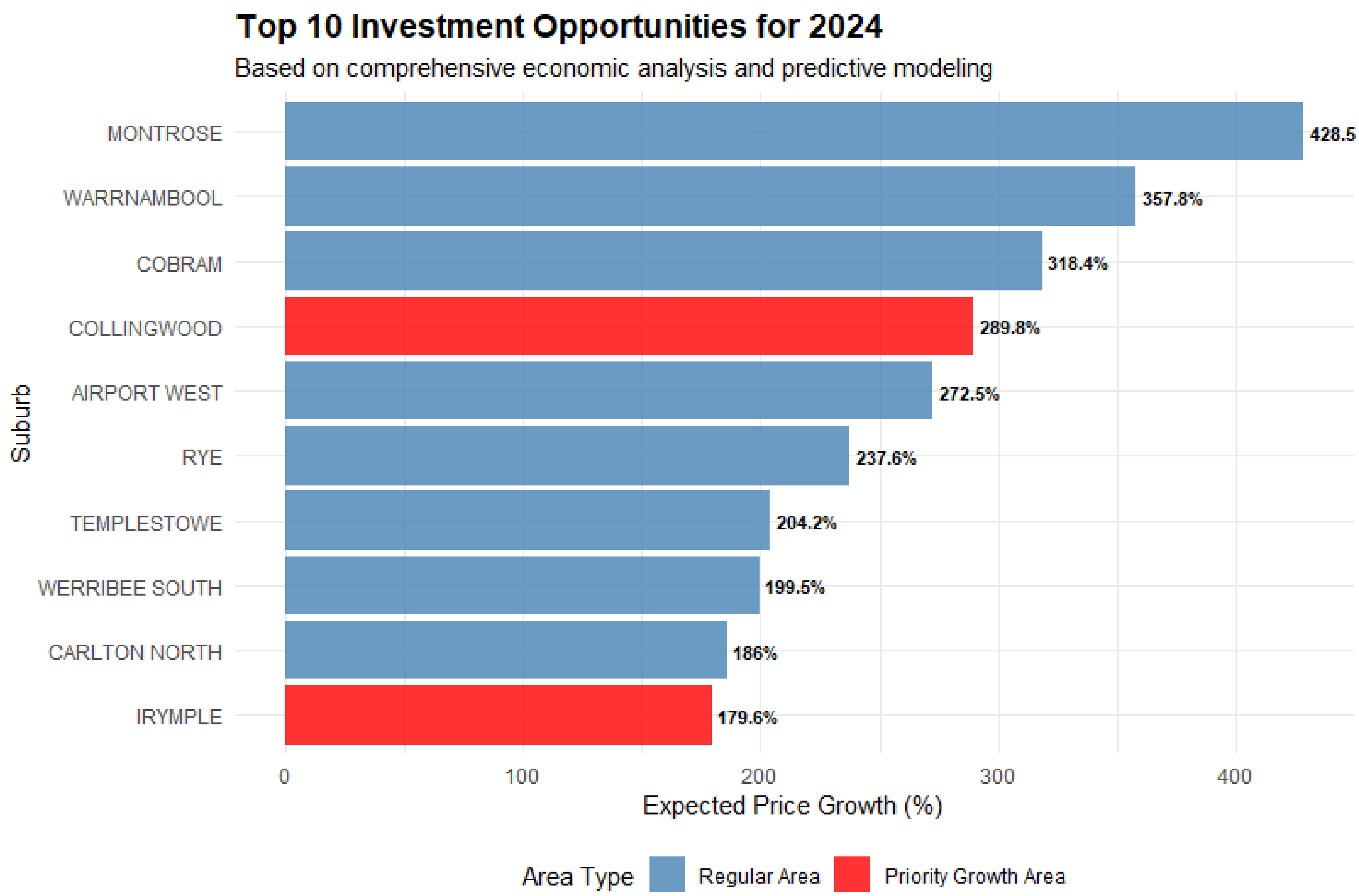
This is the linear regression equation for model_2. Model_1 couldn't account for the correlation between the other variables and thus performed slightly worse.

Diagnostic plots were constructed to check for the residuals of the model and if there were any extreme outliers, which would have been caught by the QQ plot. There seems to be no significant outliers. According to the performance of model_2, high population growth and low unemployment rates cause for the suburbs to have higher median prices. This is more true for non priority areas.



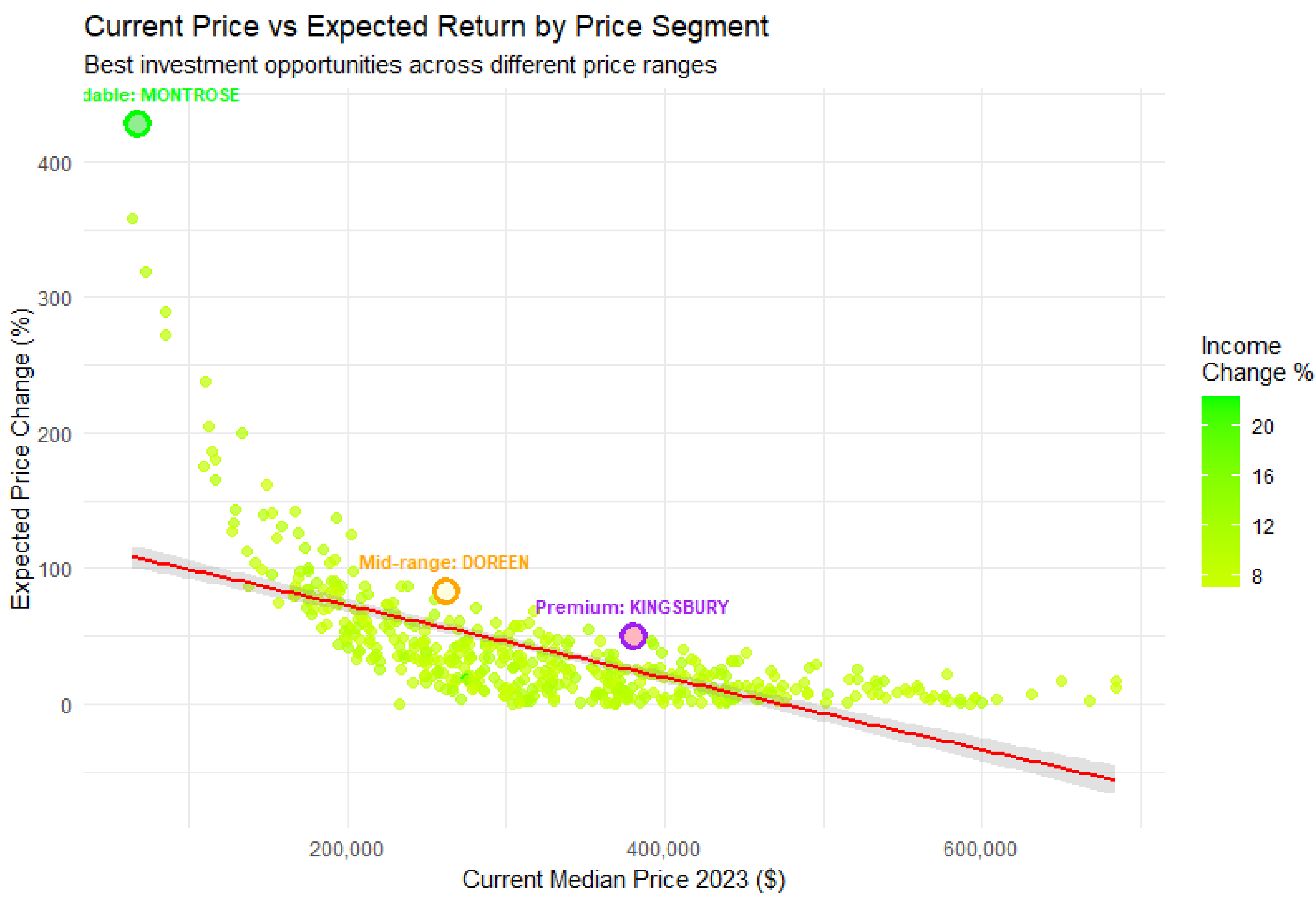
3. Model Interpretation

According to the the graphs below, I believe MONTROSE is the best suburb to invest I 2024 due to 428% increase in price




```
> cat(paste(output_lines, collapse = "\n"))
```

Rank	Suburb	2023 Price	2024 Predicted	Price Change	Price %	Income %	Unemploy. %	Pop. Growth %	Priority
1	MONTROSE	$ 67,157	$354,912	$287,755	428.5%	9.6%	5.9%	18.8%	No
2	WERRIBEE SOUTH	$133,663	$400,320	$266,657	199.5%	10.3%	5.8%	18.8%	No
3	CROYDON	$192,732	$457,299	$264,567	137.3%	10.2%	11%	19.5%	No
4	RYE	$110,376	$372,638	$262,262	237.6%	9.3%	12.4%	18.8%	No
5	GLEN IRIS	$202,528	$456,793	$254,265	125.5%	8.4%	7.6%	20.1%	No
6	COLLINGWOOD	$ 85,746	$334,213	$248,467	289.8%	9.5%	9.8%	19.3%	Yes
7	WALLAN	$148,830	$389,094	$240,264	161.4%	7.2%	12.2%	17.7%	No
8	COBURG	$166,435	$403,303	$236,868	142.3%	8.9%	13.6%	16.3%	No
9	AIRPORT WEST	$ 85,604	$318,902	$233,298	272.5%	9.1%	6.1%	20.5%	



Accoridngto

---

title: "EDA"

output: html_document

Importing Libraries

```{r}

library(ggplot2)

library(tidyverse)
```

library(plotly)
```

## Importing Datasets from csv

```{r}
# Load the datasets
apartments_data <- read.csv("data/Apartment_prices.csv")
historical_data <- read.csv("data/Historical_demographic.csv")
projected_data <- read.csv("data/Projected_demographic.csv")
```

Cleaning Data

```{r}
# Display basic information about datasets
cat("=== APARTMENT PRICES DATASET ===\n")
str(apartments_data)
summary(apartments_data)

```

```{r}
cat("\n=== HISTORICAL DEMOGRAPHIC DATASET ===\n")
str(historical_data)
summary(historical_data)
```

```{r}

cat("\n=== PROJECTED DEMOGRAPHIC DATASET ===\n")

str(projected_data)
summary(projected_data)
```

Missing Values

```{r}
# Check for missing values
cat("Missing values in apartments_data:\n")
sapply(apartments_data, function(x) sum(is.na(x)))

cat("Missing values in historical_data:\n")
sapply(historical_data, function(x) sum(is.na(x)))

cat("Missing values in historical_data:\n")
sapply(projected_data, function(x) sum(is.na(x)))
```

```{r}
# Check for duplicates
cat("\nDuplicate rows:\n")
cat("Apartment prices:", sum(duplicated(apartments_data)), "\n")
cat("Historical demographic:", sum(duplicated(historical_data)), "\n")
cat("Projected demographic:", sum(duplicated(projected_data)), "\n")


```

```{r}
# Get unique values in a column
suburbs_prices  <- unique(apartments_data$Suburb_name)

# Get unique values in a column
suburbs_hist  <- unique(historical_data$Suburb_name)

# Get unique values in a column
unique_values <- unique(apartments_data$Suburb_name)

# Get unique values in a column
suburbs_proj  <- unique(projected_data$Suburb_name)

length(suburbs_prices)  # Number of unique suburbs in prices
length(suburbs_hist)    # Number of unique suburbs in historical demographics
length(suburbs_proj)    # Number of unique suburbs in projected demographics

# Intersect of all three datasets
common_suburbs <- Reduce(intersect, list(suburbs_prices, suburbs_hist, suburbs_proj))
length(common_suburbs)  # Number of suburbs present in all datasets

```

Merging datasets

```{r}
# Join historical and projected demographics first
demo_combined <- left_join(historical_data, projected_data, by = "Suburb_name")

# Then merge with apartment prices
final_data <- left_join(apartments_data, demo_combined, by = "Suburb_name")

# View merged data
head(final_data)
```

```{r}
cat("Missing values in final data:\n")
sapply(final_data, function(x) sum(is.na(x)))
```

Filling missing values

```{r}
final_data$Historical_median_income[is.na(final_data$Historical_median_income)] <-
mean(final_data$Historical_median_income, na.rm = TRUE)

```

Changing datatypes

```{r}
final_data$Median_price_2023  <- parse_number(final_data$Median_price_2023 )

# Convert directly (will produce NA for non-numeric values)
final_data$Median_price_2023     <- as.numeric(final_data$Median_price_2023     )

# Check for conversion issues
sum(is.na(final_data$Median_price_2023     )) > sum(is.na(final_data$Median_price_2023     ))
```

```{r}
colnames(final_data)
```
```

````
```{r}
final_data %>%
  select_if(is.numeric) %>%
  summary()
```

```{r}
# Using sapply (base R)
unique_counts <- sapply(final_data, function(x) length(unique(x)))
print(unique_counts)



```

## Outlier detection

```{r}

price_outliers <- final_data %>%
  filter(!is.na(Median_price_2023)) %>%
  mutate(
    Q1 = quantile(Median_price_2023, 0.25, na.rm = TRUE),
    Q3 = quantile(Median_price_2023, 0.75, na.rm = TRUE),
    IQR = Q3 - Q1,
    lower_bound = Q1 - 1.5 * IQR,
    upper_bound = Q3 + 1.5 * IQR,
    is_outlier = Median_price_2023 < lower_bound | Median_price_2023 > upper_bound
  )

cat("\n=== PRICE OUTLIERS ===\n")
````
````

price_outliers %>%
  filter(is_outlier) %>%
  select(Suburb_name, Median_price_2023) %>%
  arrange(desc(Median_price_2023)) %>%
  print()

```

```{r}
# Replace all values in 'MAFFRA' with the column median
final_data <- final_data %>%
  mutate(Median_price_2023 = ifelse(
    Suburb_name == "MAFFRA",
    median(Median_price_2023, na.rm = TRUE),
    Median_price_2023
  ))
```

```{r}
# Converting negative to positive for unemployment rate
final_data$Historical_unemployment_rate <- abs(final_data$Historical_unemployment_rate)

final_data$Projected_unemployment_rate <- abs(final_data$Projected_unemployment_rate)
```

Saving cleaned csv

```{r}
write.csv(final_data, file = "data/merged_cleaned_data.csv", row.names = FALSE)

```
```

```{r}

## Get all numeric columns
cols_to_exclude <- c("Projected_population_growth", "Projected_median_income",
                    "Projected_unemployment_rate",
"Projected_priority_growth_area","Historical_priority_growth_area")
numeric_cols <- setdiff(names(final_data)[sapply(final_data, is.numeric)], cols_to_exclude)



for (col in numeric_cols) {
  # Calculate statistics for normal curve
  col_mean <- mean(final_data[[col]], na.rm = TRUE)
  col_sd <- sd(final_data[[col]], na.rm = TRUE)

  # Create histogram with normal curve
  p <- ggplot(final_data, aes(x = .data[[col]])) +
    geom_histogram(aes(y = ..density..),
                   bins = 30,
                   fill = "skyblue",  # Dodger blue
                   color = "black") +
    stat_function(fun = dnorm,
                   args = list(mean = col_mean, sd = col_sd),
                   color = "red",  # Orange red
                   size = 1) +
    labs(title = paste("Distribution of", col),
         subtitle = paste("Mean =", round(col_mean, 2),
                          "SD =", round(col_sd, 2)),
         x = col,
         y = "Density") +
    theme_minimal() +
```

```
    theme(plot.title = element_text(face = "bold", size = 12),
          plot.subtitle = element_text(size = 10))

  # Print the plot
  print(p)
}
```

Correlation analysis

```{r}
historical_numeric_vars <- final_data %>%
  select(
    Median_price_2023,
    Historical_population_growth,
    Historical_median_income,
    Historical_unemployment_rate
    # Add other historical numeric variables as needed
  )


correlation_matrix <- cor(historical_numeric_vars, use = "complete.obs")


# Plot correlation matrix
corrplot(correlation_matrix, method = "color", type = "upper",
         tl.cex = 0.8, tl.col = "black", tl.srt = 45)


correlation_matrix
```

```{r}
x_var <- "Historical_unemployment_rate"
y_var <- "Historical_population_growth"

ggplot(final_data, aes(x = .data[[x_var]], y = .data[[y_var]])) +
  geom_point(color = "brown") +
  geom_smooth(method = "lm", se = FALSE)+
  labs(title = paste(x_var, "vs", y_var),
       x = x_var,
       y = y_var) +
  theme_minimal()




```

```{r}
z_var <- "Historical_priority_growth_area"  # Your binary column (TRUE/FALSE or 0/1)

ggplot(final_data, aes(
  x = .data[[x_var]],
  y = .data[[y_var]],
  color = as.factor(.data[[z_var]])  # Convert to factor for discrete colors
)) +
  geom_point( alpha = 0.7) +
  scale_color_manual(
    values = c("0" = "blue", "1" = "red"),  # Custom colors
    labels = c("Non-Priority", "Priority"),  # Legend labels
    name = "Priority Area"                   # Legend title
```

````
  ) +


  geom_smooth(method = "lm", se = FALSE)+
  labs(
    title = paste(x_var, "vs", y_var, "by", z_var),
    x = x_var,
    y = y_var
  ) +
  theme_minimal() +
  theme(legend.position = "bottom")
```


```{r}


# Simple boxplot for Median_price_2023
boxplot(final_data$Median_price_2023,
        main = "Boxplot of Median Apartment Prices",
        ylab = "Price (AUD)",
        col = "lightblue")


```


```{r}
# Priority growth area vs apartment price
# Compare prices in government-prioritized growth suburbs.
ggplot(final_data, aes(x = factor(Historical_priority_growth_area), y = Median_price_2023)) +
  geom_boxplot(fill = c("tomato", "seagreen")) +
  labs(title = "Impact of Priority Growth Area on Prices",
       x = "Priority Growth Area (0 = No, 1 = Yes)",
       y = "Median Price")
````

````
```

```{r}
# Top 10 expensive
top_exp <- final_data %>% arrange(desc(Median_price_2023)) %>% slice(1:10)

ggplot(top_exp, aes(x = reorder(Suburb_name, Median_price_2023), y = Median_price_2023)) +
  geom_col(fill = "firebrick") +
  coord_flip() +
  labs(title = "Top 10 Most Expensive Suburbs (2023)", x = "Suburb", y = "Median Price")

# Top 10 affordable
bottom_exp <- final_data %>% arrange(Median_price_2023) %>% slice(1:10)

ggplot(bottom_exp, aes(x = reorder(Suburb_name, -Median_price_2023), y = Median_price_2023)) +
  geom_col(fill = "forestgreen") +
  coord_flip() +
  labs(title = "Top 10 Most Affordable Suburbs (2023)", x = "Suburb", y = "Median Price")


```

```{r}
#Shows if wealthier areas correlate with higher apartment prices.
ggplot(final_data, aes(x = Historical_median_income, y = Median_price_2023)) +
  geom_point(color = "navy") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Apartment Price vs Median Income", x = "Median Income", y = "Median Price")

```
````
````

````
```{r}
# Can reveal if fast-growing suburbs have undervalued properties.
ggplot(final_data, aes(x = Projected_population_growth, y = Median_price_2023)) +
  geom_point(color = "darkorange") +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  labs(title = "Projected Pop Growth vs Current Apartment Price")


```


```{r}
#Inverse trends may suggest distressed opportunities.
ggplot(final_data, aes(x = Projected_unemployment_rate, y = Median_price_2023)) +
  geom_point(color = "purple") +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  labs(title = "Unemployment Rate vs Median Apartment Price")




```


```{r}



# Select top 20 suburbs by projected population growth
top_growth <- final_data %>%
  arrange(desc(Historical_population_growth)) %>%
  slice(1:20)

# Create interactive plot
plot_ly(
````
````

```
  data = top_growth,
  x = ~Historical_population_growth,
  y = ~Median_price_2023,
  type = 'scatter',
  mode = 'markers+text',
  text = ~Suburb_name,
  textposition = 'top center',
  marker = list(
    size = 12,
    color = ~Median_price_2023,
    colorscale = 'Viridis',
    showscale = TRUE,
    colorbar = list(title = "Median Price")
  )
) %>%
  layout(
    title = list(
      text = "Top 20 Suburbs by Historical Population Growth<br><sup>Colored by 2023 Median Apartment Price</sup>",
      x = 0.1
    ),
    xaxis = list(title = "Historical Population Growth (%)"),
    yaxis = list(title = "Median Apartment Price (2023)"),
    font = list(family = "Arial", size = 12)
  )

```

```r
library(tidyverse)
library(ggplot2)
library(gridExtra)

# Load data
data <- read_csv("data/merged_cleaned_data.csv")



# ============================================================================
# PART 1: LINEAR REGRESSION MODELS AND COMPARISON
# ============================================================================


# Fit the four models
model_1 <- lm(Median_price_2023 ~
        Historical_population_growth +
        Historical_median_income +
        Historical_unemployment_rate +
        Historical_priority_growth_area,
       data = data)



model_2 <- lm(Median_price_2023 ~
        poly(Historical_population_growth, 2) +
        Historical_unemployment_rate +
        Historical_priority_growth_area +
        Historical_unemployment_rate:Historical_population_growth +
        Historical_population_growth:Historical_priority_growth_area,
```

```
              data = data)



# Print model summaries
cat("MODEL SUMMARIES\n")
cat("===============\n\n")

cat("MODEL 1 - Linear Model:\n")
print(summary(model_1))
cat("\n", rep("=", 50), "\n\n")

cat("MODEL 2 - :\n")
print(summary(model_2))
cat("\n", rep("=", 50), "\n\n")



# Calculate model performance metrics
calculate_metrics <- function(model) {
  n <- nobs(model)
  k <- length(coef(model))

  # R-squared and Adjusted R-squared
  r_squared <- summary(model)$r.squared
  adj_r_squared <- summary(model)$adj.r.squared

  # AIC and BIC
  aic <- AIC(model)
  bic <- BIC(model)
```

```
  # RMSE
  rmse <- sqrt(mean(residuals(model)^2))

  # Residual standard error
  rse <- summary(model)$sigma

  return(data.frame(
    R2 = r_squared,
    Adj_R2 = adj_r_squared,
    AIC = aic,
    BIC = bic,
    RMSE = rmse,
    RSE = rse
  ))
}

# Create performance comparison table
model_metrics <- rbind(
  calculate_metrics(model_1),
  calculate_metrics(model_2)
)

model_metrics$Model <- c("Model 1 (Linear)",
                         "Model 2 (Quadratic)")

# Reorder columns
model_metrics <- model_metrics[, c("Model", "R2", "Adj_R2", "AIC", "BIC", "RMSE", "RSE")]

# Print performance table
cat("MODEL PERFORMANCE COMPARISON\n")
cat("============================\n\n")
```

```
print(model_metrics, row.names = FALSE, digits = 4)



# Create fitted vs actual plots
create_fitted_actual_plot <- function(model, title) {
  actual <- model$model[[1]]
  predicted <- fitted(model)

  plot_data <- data.frame(
    Actual = actual,
    Predicted = predicted
  )

  # Calculate R-squared for the plot
  r_squared <- cor(plot_data$Actual, plot_data$Predicted)^2

  ggplot(plot_data, aes(x = Actual, y = Predicted)) +
    geom_point(alpha = 0.6, color = "steelblue", size = 1.5) +
    geom_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed", size = 1) +
    geom_smooth(method = "lm", se = TRUE, color = "darkgreen", alpha = 0.3) +
    labs(title = paste(title, "- Fitted vs Actual"),
         x = "Actual Median Price 2023",
         y = "Predicted Median Price 2023") +
    scale_x_continuous(labels = scales::comma) +
    scale_y_continuous(labels = scales::comma) +
    theme_minimal() +
    theme(
      plot.title = element_text(size = 12, face = "bold"),
      panel.grid.minor = element_blank()
    ) +
    annotate("text", x = min(plot_data$Actual) + 0.1 * diff(range(plot_data$Actual)),
```

```
      y = max(plot_data$Predicted) - 0.1 * diff(range(plot_data$Predicted)),
      label = paste("R² =", round(r_squared, 3)),
      size = 4, fontface = "bold", color = "darkblue")
}

# Create fitted vs actual plots for all models
p2 <- create_fitted_actual_plot(model_1, "Model 1")
p3 <- create_fitted_actual_plot(model_2, "Model 2")

# Create diagnostic plots
create_diagnostic_plots <- function(model, title) {
  # Get residuals and fitted values
  residuals <- residuals(model)
  fitted_vals <- fitted(model)
  standardized_residuals <- rstandard(model)

  # Create diagnostic data frame
  diag_data <- data.frame(
    fitted = fitted_vals,
    residuals = residuals,
    std_residuals = standardized_residuals,
    sqrt_abs_std_resid = sqrt(abs(standardized_residuals))
  )

  # 1. Residuals vs Fitted
  p1 <- ggplot(diag_data, aes(x = fitted, y = residuals)) +
    geom_point(alpha = 0.6, color = "steelblue") +
    geom_smooth(method = "loess", se = FALSE, color = "red") +
    geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
    labs(title = "Residuals vs Fitted", x = "Fitted Values", y = "Residuals") +
    theme_minimal() +
```

```r
  theme(plot.title = element_text(size = 11, face = "bold"))

 # 2. Q-Q Plot
 p2 <- ggplot(diag_data, aes(sample = std_residuals)) +
  stat_qq(alpha = 0.6, color = "steelblue") +
  stat_qq_line(color = "red") +
  labs(title = "Q-Q Plot", x = "Theoretical Quantiles", y = "Standardized Residuals") +
  theme_minimal() +
  theme(plot.title = element_text(size = 11, face = "bold"))

 # 3. Scale-Location
 p3 <- ggplot(diag_data, aes(x = fitted, y = sqrt_abs_std_resid)) +
  geom_point(alpha = 0.6, color = "steelblue") +
  geom_smooth(method = "loess", se = FALSE, color = "red") +
  labs(title = "Scale-Location", x = "Fitted Values", y = "√|Standardized Residuals|") +
  theme_minimal() +
  theme(plot.title = element_text(size = 11, face = "bold"))

 # 4. Residuals vs Leverage (simplified)
 p4 <- ggplot(diag_data, aes(x = seq_along(residuals), y = std_residuals)) +
  geom_point(alpha = 0.6, color = "steelblue") +
  geom_smooth(method = "loess", se = FALSE, color = "red") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  labs(title = "Residuals vs Order", x = "Observation Order", y = "Standardized Residuals") +
  theme_minimal() +
  theme(plot.title = element_text(size = 11, face = "bold"))

 # Combine plots
 grid.arrange(p1, p2, p3, p4, ncol = 2, top = paste("Diagnostic Plots -", title))
}
```

```r
# Show fitted vs actual plots
grid.arrange(p2, p3,  ncol = 2, top = "Fitted vs Actual Values")

# Show diagnostic plots for each model
create_diagnostic_plots(model_1, "Model 1")
create_diagnostic_plots(model_2, "Model 2")



# Create ranking based on metrics
ranking_data <- model_metrics %>%
  mutate(
    Rank_R2 = rank(-Adj_R2),
    Rank_AIC = rank(AIC),
    Rank_BIC = rank(BIC),
    Rank_RMSE = rank(RMSE),
    Avg_Rank = (Rank_R2 + Rank_AIC + Rank_BIC + Rank_RMSE) / 4
  ) %>%
  arrange(Avg_Rank) %>%
  select(Model, Adj_R2, AIC, BIC, RMSE, Avg_Rank)

cat("\n\nMODEL RANKING\n")
cat("=============\n")
print(ranking_data, row.names = FALSE, digits = 4)

best_model <- ranking_data$Model[1]
cat("\n🏆 BEST MODEL:", best_model, "\n")

# Select the best model for predictions
if(grepl("Model 1", best_model)) {
```

```
  selected_model <- model_1
} else if(grepl("Model 2", best_model)) {
  selected_model <- model_2
} else if(grepl("Model 3", best_model)) {
  selected_model <- model_3
} else {
  selected_model <- model_4
}

# Create future prediction data using projected values
future_data <- data %>%
  select(Suburb_name, Median_price_2023,
         Historical_population_growth = Projected_population_growth,
         Historical_median_income = Projected_median_income,
         Historical_unemployment_rate = Projected_unemployment_rate,
         Historical_priority_growth_area = Projected_priority_growth_area)

# Make predictions
predictions <- predict(selected_model, newdata = future_data)

# Create results dataframe
results <- data.frame(
  Suburb_name = data$Suburb_name,
  Current_Price_2023 = data$Median_price_2023,
  Predicted_Future_Price = predictions,
  Price_Growth = predictions - data$Median_price_2023,
  Growth_Percentage = ((predictions - data$Median_price_2023) / data$Median_price_2023) * 100
) %>%
  arrange(desc(Predicted_Future_Price))
```

```
# Best suburb recommendation
best_suburb <- results[1, ]
cat("\n\n🏆 RECOMMENDED INVESTMENT SUBURB:", best_suburb$Suburb_name, "\n")
cat("Current Median Price (2023): $", format(best_suburb$Current_Price_2023, big.mark = ","), "\n")
cat("Predicted Future Price: $", format(best_suburb$Predicted_Future_Price, big.mark = ","), "\n")
cat("Expected Price Growth: $", format(best_suburb$Price_Growth, big.mark = ","), "\n")
cat("Expected Growth Rate: ", round(best_suburb$Growth_Percentage, 2), "%\n")

# Create visualization showing top 15 suburbs
top_15 <- head(results, 15)

# 1. Bar chart of predicted prices
p1 <- ggplot(top_15, aes(x = reorder(Suburb_name, Predicted_Future_Price), y = Predicted_Future_Price)) +
  geom_col(fill = "steelblue", alpha = 0.7) +
  geom_text(aes(label = paste0("$", round(Predicted_Future_Price/1000, 0), "K")),
            hjust = -0.1, size = 3, fontface = "bold") +
  coord_flip() +
  labs(title = "Top 15 Suburbs by Predicted Future Median Price",
       subtitle = paste("Based on", best_model),
       x = "Suburb", y = "Predicted Future Median Price ($)") +
  scale_y_continuous(labels = scales::comma) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 11),
    axis.text.y = element_text(size = 9)
  )

# 2. Growth comparison chart
p2 <- ggplot(top_15, aes(x = reorder(Suburb_name, Price_Growth), y = Price_Growth)) +
```

```r
  geom_col(fill = "darkgreen", alpha = 0.7) +
  geom_text(aes(label = paste0("$", round(Price_Growth/1000, 0), "K")),
            hjust = -0.1, size = 3, fontface = "bold") +
  coord_flip() +
  labs(title = "Top 15 Suburbs by Expected Price Growth",
       subtitle = "Absolute dollar growth from current to predicted prices",
       x = "Suburb", y = "Expected Price Growth ($)") +
  scale_y_continuous(labels = scales::comma) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 11),
    axis.text.y = element_text(size = 9)
  )


print(p1)
print(p2)


cat("SUBURBAN PROPERTY INVESTMENT ANALYSIS\n")
cat("=====================================\n\n")
```

```
# ============================================================================
# PART 2: PREDICT 2024 VALUES AND CALCULATE PERCENTAGE CHANGES
# ============================================================================


# Create 2024 prediction data with improved economic conditions
data_2024 <- data %>%
  mutate(
    # 2024 projections: improved economic conditions
    Predicted_median_income_2024 = Projected_median_income * 1.05,  # 5% income increase
    Predicted_unemployment_rate_2024 = Projected_unemployment_rate * 0.9,  # 10% unemployment
decrease
    Predicted_population_growth_2024 = Projected_population_growth * 1.15,  # 15% population growth
increase
    Predicted_priority_growth_area_2024 = Projected_priority_growth_area  # Same as projected
  )

# Use the best model to predict 2024 median prices
# Map the variable names to match the model
prediction_data <- data_2024 %>%
  select(Suburb_name, Median_price_2023,
         Historical_population_growth = Predicted_population_growth_2024,
         Historical_median_income = Predicted_median_income_2024,
         Historical_unemployment_rate = Predicted_unemployment_rate_2024,
         Historical_priority_growth_area = Predicted_priority_growth_area_2024)

# Make predictions
predictions_2024 <- predict(selected_model, newdata = prediction_data)

# Calculate percentage changes and create comprehensive results
investment_analysis <- data_2024 %>%
  mutate(
```

```r
    Predicted_median_price_2024 = predictions_2024,

    # Price changes
    Price_change = Predicted_median_price_2024 - Median_price_2023,
    Price_change_percent = abs((Predicted_median_price_2024 - Median_price_2023) / Median_price_2023) *
100,

    # Economic indicator changes
    Income_change_percent = ((Predicted_median_income_2024 - Historical_median_income) /
Historical_median_income) * 100,
    Income_change = (Predicted_median_income_2024 - Historical_median_income) ,
    Unemployment_change_percent = ((Predicted_unemployment_rate_2024 - Historical_unemployment_rate)
/ Historical_unemployment_rate) * 100,
    Unemployment_change = (Predicted_unemployment_rate_2024 - Historical_unemployment_rate),
    Population_growth_change_percent =((Predicted_population_growth_2024 -
Historical_population_growth) / Historical_population_growth) * 100,
    Population_growth_change = (Predicted_population_growth_2024 - Historical_population_growth)

  ) %>%
  arrange(desc(Population_growth_change_percent))

# Display comprehensive investment table
cat("COMPREHENSIVE INVESTMENT ANALYSIS TABLE\n")
cat("========================================\n\n")

# Create a properly formatted top 30 suburbs table with multiple sorting criteria
# Sort by Price_change (desc), Population_growth_change (desc), Unemployment_change (desc)
investment_sorted <- investment_analysis[order(-investment_analysis$Price_change,
                                               -investment_analysis$Population_growth_change,
```

```
-investment_analysis$Unemployment_change), ]

top_30_suburbs <- investment_sorted[1:30, ]

# Create a clean data frame for display
display_table <- data.frame(
  Rank = 1:30,
  Suburb = top_30_suburbs$Suburb_name,
  Current_Price_2023 = format(round(top_30_suburbs$Median_price_2023, 0), big.mark = ","),
  Predicted_Price_2024 = format(round(top_30_suburbs$Predicted_median_price_2024, 0), big.mark = ","),
  Price_change = format(round(top_30_suburbs$Price_change, 0), big.mark = ","),
  Price_Growth_Pct = round(top_30_suburbs$Price_change_percent, 1),
  Income_Growth_Pct = round(top_30_suburbs$Income_change_percent, 1),
  Unemployment_Change_Pct = round(top_30_suburbs$Unemployment_change_percent, 1),
  Population_Growth_Change_Pct = round(top_30_suburbs$Population_growth_change_percent, 1),
  Priority_Area = ifelse(top_30_suburbs$Predicted_priority_growth_area_2024 == 1, "Yes", "No")
)


output_lines <- c()
# Print column headers
header <- sprintf("%-4s %-20s %-15s %-17s %-14s %-12s %-12s %-15s %-22s %-10s",
                  "Rank", "Suburb", "2023 Price", "2024 Predicted", "Price Change",
                  "Price %", "Income %", "Unemploy. %", "Pop. Growth %", "Priority")
output_lines <- c(output_lines, header)
output_lines <- c(output_lines, paste(rep("-", 150), collapse = ""))

for (i in 1:nrow(display_table)) {
  row_line <- sprintf("%-4d %-20s %-15s %-17s %-14s %-12s %-12s %-15s %-22s %-10s",
                      display_table$Rank[i],
                      substr(display_table$Suburb[i], 1, 20),
```

```
                  paste0("$", display_table$Current_Price_2023[i]),
                  paste0("$", display_table$Predicted_Price_2024[i]),
                  paste0("$", display_table$Price_change[i]),
                  paste0(display_table$Price_Growth_Pct[i], "%"),
                  paste0(display_table$Income_Growth_Pct[i], "%"),
                  paste0(display_table$Unemployment_Change_Pct[i], "%"),
                  paste0(display_table$Population_Growth_Change_Pct[i], "%"),
                  display_table$Priority_Area[i]
  )
  output_lines <- c(output_lines, row_line)
}

# Print all at once
cat(paste(output_lines, collapse = "\n"))
# ===========================================================================
# PART 3: INDIVIDUAL ANALYSIS SECTIONS WITH PLOTS
# ===========================================================================

cat("\n\nINDIVIDUAL ANALYSIS SECTIONS\n")
cat("============================\n\n")

# Section 1: Price Growth Analysis by Suburb
cat("1. PRICE GROWTH ANALYSIS BY SUBURB\n")
cat("-----------------------------------\n")

top_15_price_growth <- head(investment_analysis, 15)

p_price_growth <- ggplot(top_15_price_growth, aes(x = reorder(Suburb_name, Price_change_percent),
                                                   y = Price_change_percent)) +
  geom_col(fill = "steelblue", alpha = 0.8) +
  geom_text(aes(label = paste0(round(Price_change_percent, 1), "%")),
```

```r
           hjust = -0.1, size = 3, fontface = "bold") +
  coord_flip() +
  labs(title = "Top 15 Suburbs by Expected Price Growth (2024)",
       subtitle = paste("Based on", best_model),
       x = "Suburb", y = "Expected Price Growth (%)") +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 11),
    axis.text.y = element_text(size = 9)
  )

print(p_price_growth)




# Section 2: Population Growth Impact
cat("\n4. POPULATION GROWTH IMPACT\n")
cat("---------------------------\n")

p_population_impact <- ggplot(investment_analysis, aes(x = Predicted_population_growth_2024,
                                                        y = Price_change_percent)) +
  geom_point(aes(color = Predicted_median_income_2024), alpha = 0.7, size = 2) +
  geom_smooth(method = "lm", se = TRUE, color = "red", alpha = 0.3) +
  scale_color_gradient(low = "lightblue", high = "darkblue", name = "Median\nIncome 2024",
                       labels = comma) +
  labs(title = "Population Growth vs Price Growth",
       subtitle = "Higher population growth drives housing demand",
       x = "Predicted Population Growth 2024 (%)",
       y = "Price Change (%)") +
```

```r
  theme_minimal()

print(p_population_impact)

# Section 5: Current Price vs Expected Return Analysis
cat("\n5. CURRENT PRICE VS EXPECTED RETURN ANALYSIS\n")
cat("-------------------------------------------\n")

# Identify best suburbs for different price ranges
best_affordable <- investment_analysis %>%
  filter(Median_price_2023 <= quantile(Median_price_2023, 0.33)) %>%
  slice_max(Price_change_percent, n = 1)

best_midrange <- investment_analysis %>%
  filter(Median_price_2023 > quantile(Median_price_2023, 0.33) &
           Median_price_2023 <= quantile(Median_price_2023, 0.67)) %>%
  slice_max(Price_change_percent, n = 1)

best_premium <- investment_analysis %>%
  filter(Median_price_2023 > quantile(Median_price_2023, 0.67)) %>%
  slice_max(Price_change_percent, n = 1)

p_price_return <- ggplot(investment_analysis, aes(x = Median_price_2023, y = Price_change_percent)) +
  geom_point(aes(color = Income_change_percent), alpha = 0.7, size = 2) +
  geom_smooth(method = "lm", se = TRUE, color = "red", alpha = 0.3) +

  # Highlight best suburbs in each category
  geom_point(data = best_affordable, aes(x = Median_price_2023, y = Price_change_percent),
             color = "green", size = 4, shape = 21, fill = "lightgreen", stroke = 2) +
  geom_point(data = best_midrange, aes(x = Median_price_2023, y = Price_change_percent),
             color = "orange", size = 4, shape = 21, fill = "lightyellow", stroke = 2) +
```

```r
  geom_point(data = best_premium, aes(x = Median_price_2023, y = Price_change_percent),
             color = "purple", size = 4, shape = 21, fill = "lightpink", stroke = 2) +

  # Add labels for best suburbs
  geom_text(data = best_affordable, aes(x = Median_price_2023, y = Price_change_percent,
                                         label = paste("Affordable:", Suburb_name)),
            vjust = -1.5, hjust = 0.5, size = 3, fontface = "bold", color = "green") +
  geom_text(data = best_midrange, aes(x = Median_price_2023, y = Price_change_percent,
                                       label = paste("Mid-range:", Suburb_name)),
            vjust = -1.5, hjust = 0.5, size = 3, fontface = "bold", color = "orange") +
  geom_text(data = best_premium, aes(x = Median_price_2023, y = Price_change_percent,
                                      label = paste("Premium:", Suburb_name)),
            vjust = -1.5, hjust = 0.5, size = 3, fontface = "bold", color = "purple") +

  scale_x_continuous(labels = comma) +
  scale_color_gradient2(low = "red", mid = "yellow", high = "green", midpoint = 0,
                        name = "Income\nChange %") +
  labs(title = "Current Price vs Expected Return by Price Segment",
       subtitle = "Best investment opportunities across different price ranges",
       x = "Current Median Price 2023 ($)",
       y = "Expected Price Change (%)") +
  theme_minimal()

print(p_price_return)

# ==========================================================================
# PART 4: FINAL INVESTMENT RECOMMENDATIONS
# ==========================================================================



# Investment summary visualization
```

```r
top_10_final <- head(investment_analysis, 10)

p_final_summary <- ggplot(top_10_final, aes(x = reorder(Suburb_name, Price_change_percent),
                                            y = Price_change_percent)) +
  geom_col(aes(fill = factor(Predicted_priority_growth_area_2024)), alpha = 0.8) +
  geom_text(aes(label = paste0(round(Price_change_percent, 1), "%")),
            hjust = -0.1, size = 3, fontface = "bold") +
  coord_flip() +
  scale_fill_manual(values = c("0" = "steelblue", "1" = "red"),
                    labels = c("Regular Area", "Priority Growth Area")) +
  labs(title = "Top 10 Investment Opportunities for 2024",
       subtitle = "Based on comprehensive economic analysis and predictive modeling",
       x = "Suburb", y = "Expected Price Growth (%)",
       fill = "Area Type") +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 11),
    axis.text.y = element_text(size = 9),
    legend.position = "bottom"
  )

print(p_final_summary)



################################################################################
model2_predictions <- fitted(model_2)
model2_residuals <- residuals(model_2)

# Create a comprehensive analysis dataframe
```

```
analysis_data <- data.frame(
  Suburb = data$Suburb_name,
  Actual_Price = data$Median_price_2023,
  Predicted_Price = model2_predictions,
  Residuals = model2_residuals,
  Pop_Growth = data$Historical_population_growth,
  Unemployment = data$Historical_unemployment_rate,
  Priority_Area = data$Historical_priority_growth_area,
  Pop_Growth_Squared = data$Historical_population_growth^2,
  Interaction_Unemp_Pop = data$Historical_unemployment_rate * data$Historical_population_growth,
  Interaction_Pop_Priority = data$Historical_population_growth * data$Historical_priority_growth_area
)

# Sort by predicted price for rankings
analysis_data <- analysis_data[order(-analysis_data$Predicted_Price), ]
# Sort by population growth for better visualization
pop_sorted <- analysis_data[order(analysis_data$Pop_Growth), ]

 #Get suburbs with largest residuals (both positive and negative)
analysis_data$Abs_Residuals <- abs(analysis_data$Residuals)
top_residuals <- head(analysis_data[order(-analysis_data$Abs_Residuals), ], 20)

# Create color coding for over/under prediction
res_colors <- ifelse(top_residuals$Residuals > 0, "green", "orange")

# Create bar plot of residuals
barplot(top_residuals$Residuals,
    names.arg = top_residuals$Suburb,
    col = res_colors,
    main = "Model 2: Residual Analysis\nTop 20 Suburbs by Prediction Error",
    ylab = "Residuals ($)",
```

```
    las = 2,  # Rotate x-axis labels
    cex.names = 0.7)

# Add horizontal line at zero
abline(h = 0, col = "black", lwd = 2)

# Add legend
legend("topright", c("Over-predicted", "Under-predicted"),
   col = c("green", "orange"), pch = 15, cex = 0.8)

# Reset plotting parameters
par(mfrow = c(1, 1))
```