# Exploring Variability in Rehoming Times: A Statistical Analysis of Dog Breeds in Shelter Environments

(Mohamed Imthiyas Abdul Rasheeth)

## 1. Introduction

Stray, unwanted, or neglected dogs are often sent to animal shelters to be rehomed. Previous research suggests that a dog takes around 27 weeks to rehome. We will investigate if the time to rehome is the same across all the species or if it changes from species to species.

The sample dataset can be downloaded here: mysample

## 2. Data Understanding

Our sample has three dog species: **Dobermann**, **Labrador Retriever**, and **Staffordshire Bull Terrier**. We find that there are 305 instances with 7 variables in our dataset. The **Population mean** is **27**, and the **Population variance** is **74**. We have the following seven columns in our dataset,

Rehomed – *Number of weeks taken for rehoming.*
Visited – *Number of weeks until the first visit from a potential new owner.*
Health – *Measurement of Dog's Physical health (0-100)*
Breed – *Dog's Breed*
Age – *Dog's Age (Puppy or Adult)*
Reason – *Sheltered Reason*
Returned – *If the dog is returned after rehoming (Yes, No)*

## 3. Data Preparation & Exploration

### 3. 1. Data Cleaning

Upon investigation, we found 20 NA values in our data. 6 are from Breed, and 14 are from Returned. We can also see nine values as "999999" instead of NA in Rehomed. In Returned, we also have 4 "Unknown" values, where we should either have "Yes" or "No" in our dataset. Since our primary goal is to find the rehoming time for each breed, we will only clean the **15** outliers from the Breed and Rehomed column, removing **4.91%** from our total sample data. The reason behind not removing all the other outliers is that if we removed them, our dataset would be biased due to less data. Now, we have **290 rows** and **7 columns** after data cleaning.

### 3. 2. Data Exploration

Our clean dataset has **21** Dobermann, **62** Labrador Retriever and **207** Staffordshire Bull Terrier. We also found some outliers in the Visited columns for Dobermann and Labrador

Retriever, whose minimum values are -2 and -1, respectively, but they cannot be negative numbers.

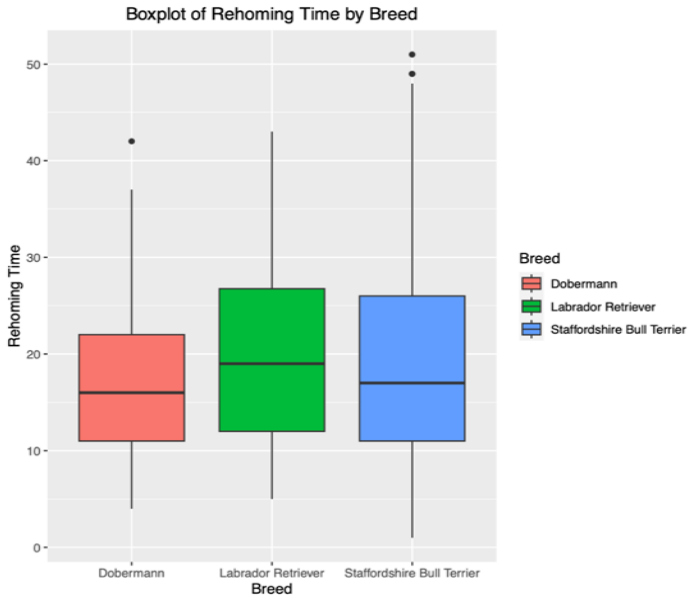


*Figure 3.1: Boxplots of Rehoming Time for Each Breed*

We split our dataset into three groups of Breeds. Figure 3.1 shows that the average rehoming times are **17.85**, **19.85** and **19.33** for Dobermann, Labrador Retriever and Staffordshire Bull Terrier, respectively.

The box plot is more spread for the Staffordshire breed because it has more samples than the other breeds. We can also see that the median and mean are closer, meaning the data are well-distributed.

## 3. 3. Data Modelling

Since we are working on Rehomed time, even though the values in Rehomed are discrete, we will assume it to be **Continuous** data. So, we will check if our sample has exponential, normal or uniform distribution. Thus, we take a QQ-Plot to see the nature of the breed.

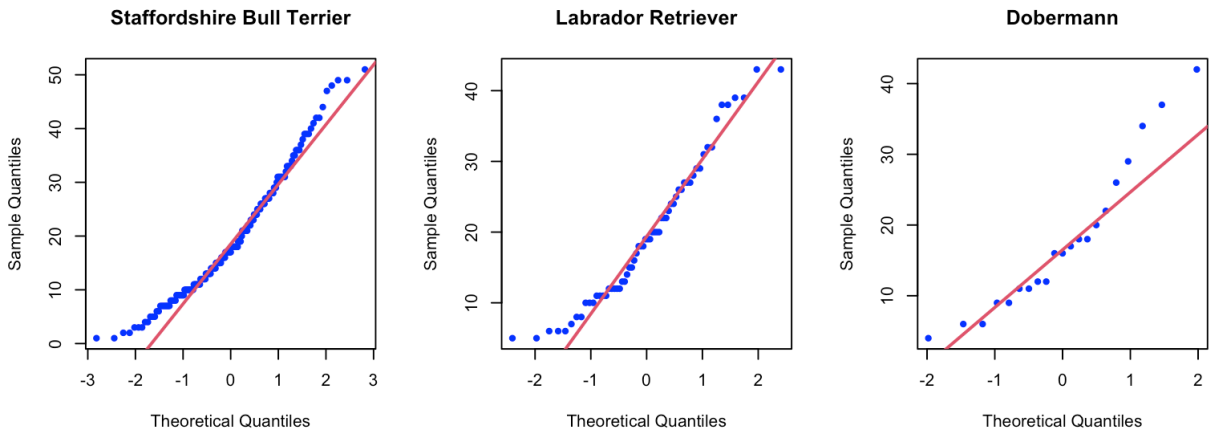


*Figure 3.2: QQ Plots of Rehoming Time for each Breed showing the distribution of the samples.*

From Figure 3.2, we can see that **none** of our breeds have a perfect normal distribution. Dobermann's QQ-Plot is randomly spread across the line because we have a significantly smaller number of samples (n=21) in the group. We further probe into taking the histogram using density instead of frequency to visualize the distribution better.

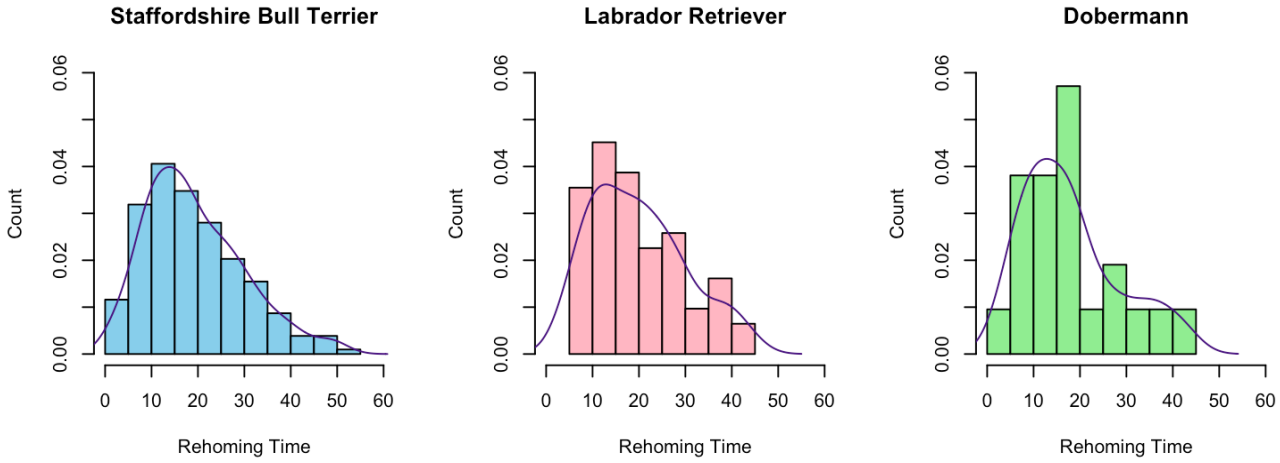


*Figure 3.3: Histogram of Rehoming Time (using density) for each Breed showing the distribution of the samples.*

As shown in Figure 3.3, we deduce from the graph that Staffordshire Bull Terrier is **Right-Skewed**, Labrador Retriever is **Light-Tailed**, and Dobermann is Fairly Normal. We cannot be sure of the normality of Dobermann because of its smaller sample size. We will then statistically test for normality since the plot shows neither exponential nor uniform distribution.

# 4. Findings & Insights

## 4. 1. Distribution

Since Rehomed is not a categorical variable, we will not use the Chi-squared (Pearson) Test. However, we chose the **Kolmogorov-Smirnov** Test for Staffordshire Bull Terrier because the sample size (n) is **greater** than 30; we also use it to test for any **specific** (normal in our case) distribution and its **ease of interpretation**. Meanwhile, for Dobermann and Labrador Retriever, as the sample size is small to moderate, and from Figure 3.3, we can confirm that it is **not a normal** distribution, we will use the **Shapiro-Wilk Test**.

(*Null*) **$H_0$: It is a Normal Distribution**
(*Alternate*) **$H_1$: It is not a Normal Distribution**

| **Breed** | **Test** | **Parameters** | **Values** |
|---|---|---|---|
| **Dobermann (n=21)** | Shapiro-Wilk Test | W Value | 0.92157 |
| | | P Value | 0.09327 |
| **Labrador Retriever (n=62)** | Shapiro-Wilk Test | W Value | 0.94918 |
| | | P Value | 0.01219 |
| **Staffordshire Bull Terrier (n=207)** | Kolmogorov-Smirnov Test | D Value | 0.11079 |
| | | P Value | 0.01242 |

*Table 4.1: Kolmogorov-Smirnov and Shapiro-Wilk Test for Normality for three breeds (Dobermann, Labrador Retriever, and Staffordshire Bull Terrier)*

From Table 4.1, since the P-value is less than 5% for Labrador Retriever and Staffordshire Bull Terrier, we can **reject** the null hypothesis because their samples do not have a Normal Distribution. Although the Kolmogorov-Smirnov Test sometimes gives us Type-2 error for large sample sizes, we can confirm the rejection of the null hypothesis by looking at the graph from Figure 3.3. Dobermann failed to reject the null hypothesis since the P-value exceeds 5%. This rejection could be because of its smaller **sample size**. However, this does not mean that it follows a normal distribution.

## 4. 2. Average Rehoming Time

Since we know the **population mean** and **standard deviation**, we will use the **Z-Test** for Labrador Retriever and Staffordshire Bull Terrier to find the confidence interval because of its moderate to **large** sample size. However, for Dobermann, it is best to use **T-test** for its **smaller** sample size. So, our hypothesis is as follows,

**$H_0$: Average Rehoming Time is 27 weeks.**
**$H_1$: Average Rehoming Time is not 27 weeks.**

| Breed | Test | Parameters | Values |
|---|---|---|---|
| **Dobermann (n=21)** | T Test | Confidence Interval | [13.05859, 22.65569] |
| | | P Value | 7.468e-04 |
| | | Mean | 17.85714 |
| **Labrador Retriever (n=62)** | Z Test | Confidence Interval | [17.71359, 21.99609] |
| | | P Value | 6.143e-11 |
| | | Mean | 19.85484 |
| **Staffordshire Bull Terrier (n=207)** | Z Test | Confidence Interval | [18.16630, 20.51003] |
| | | P Value | 2.2e-16 |
| | | Mean | 19.33816 |

*Table 4.2: Using Z and T-tests, Mean and confidence Interval for each breed (Dobermann, Labrador Retriever, and Staffordshire Bull Terrier)*

We can reject our null hypothesis since P-values are less than 5% in Table 4.2. We also deduced the confidence intervals for each breed, whose average means are 17.85, 19.85 and 19.33. These values estimate that the average rehoming time **changes** from breed to breed.

## 4. 3. Comparison of Breed

We will now compare the two breeds to check for similarities in rehoming time. Since we are dealing with various sample sizes, we will use the T-Test for this specific **flexibility** and to make our tests more consistent. Below is our hypothesis,

**$H_0$: No difference in Average Rehoming Time**
**$H_1$: A difference in Average Rehoming Time**

| Breed | Confidence Interval | P Value (Using T Test) |
|---|---|---|
| **Dobermann vs. Labrador Retriever** | [-7.143958, 3.148566] | 0.4421466 |
| **Dobermann vs. Staffordshire Bull Terrier** | [-6.247444, 3.285402] | 0.5409693 |
| **Labrador Retriever vs. Staffordshire Bull Terrier** | [-2.467626, 3.500975] | 0.7334655 |

*Table 4.3: Comparisons in Average Rehoming Time between breeds using T Test*

Table 4.3 shows that the P-values exceed 5%, and the confidence intervals include **zero** and are **narrow**, which cannot reject the null hypothesis. Hence, these results suggest **no statistical difference** in the mean of the rehoming time.

## 5. Discussion

From the above analysis, we can deduce that the samples do not follow Normal, Exponential, Uniform, Poisson, Binomial or Geometric distribution. We have also tried transforming the data into a **logarithmic** function to check for normality, which also failed to prove the normality. However, we found **Weibull** Distribution to fit our sample upon further research. However, knowing the **Population mean** and **standard deviation** made us narrow down our analysis of the distribution. We also **assumed** our sample to be of normal distribution to do the Shapiro-Wilk and Z-tests, which is seen as a limitation.

The average rehoming time differs from breed to breed; it is **not 27 weeks** for every breed. This time frame is also accurate in real-world experience, where people get more attracted to certain breeds over others, and thus, some breeds get easily picked.

During our analysis, we also found some outliers like the minimum values in the Visited column, 4 “Unknown” values and 14 NA values in Returned. We can remove these and further investigate the other aspects of our data, such as the **health** and **age** of the dogs, as future work.

## 6. Conclusion

We conclude from our analyses that the rehoming time is not 27 weeks for every breed. We also find that our samples do not have a normal distribution, and the average rehoming time for any two breeds has no massive difference in comparison.

## References

CRAN R. 2023. BSDA: Basic Statistics and Data Analysis. [Online]. [Accessed 14 December 2023]. Available from: https://CRAN.R-project.org/package=BSDA

CRAN R. 2023. dplyr: A Grammar of Data Manipulation. [Online]. [Accessed 14 December 2023]. Available from: https://CRAN.R-project.org/package=dplyr

CRAN R. 2015. nortest: Tests for Normality. [Online]. [Accessed 14 December 2023]. Available from: https://CRAN.R-project.org/package=nortest

Statistics How To. 2023. Weibull Distribution and Weibull Analysis. [Online]. [Accessed 14 December 2023]. Available from: https://www.statisticshowto.com/weibull-distribution/

Wickham H. 2016. ggplot2: Elegant Graphics for Data Analysis [Online]. New York: Springer-Verlag. [Accessed 14 December 2023]. Available from: https://ggplot2.tidyverse.org

W. N. Venables and B. D. Ripley. 2002. Modern Applied Statistics with S. [Online]. 4th ed. New York: Springer. [Accessed 14 December 2023]. Available from: https://www.stats.ox.ac.uk/pub/MASS4/

## Appendix (Codes in R)

```
#-----------------------------------------Introduction-----------------------------------------
load("…/mysample.RData")
Data = get('mysample')
print(Data)
print(dim(Data)) #305 rows and 7 columns
Population_Mean = 27
Population_Variance = 74
Population_Standard_Deviation = sqrt(Population_Variance)

#-------------------------------------Data Understanding-------------------------------------
#Load the necessary libraries
install.packages("dplyr")
library(dplyr)
library(ggplot2)
library(BSDA)
library(nortest)
library(MASS)

#Finding Outliers: Missing Values or NA
print(summary(Data))
#We have 6 NA in Breed; Max in Rehomed column is 99999 (These are missing values)

MissingValues = colSums(is.na(Data))
print(MissingValues) #We have 6 NA in Breed and 14 in Returned

copyData <- Data
copyData$count_na <- rowSums(is.na(Data))
print(sum(copyData$count_na)) #20 NA in total

unique_Rehomed <- unique(Data$Rehomed) # Get unique values in the Rehomed column
print(unique_Rehomed)
```

```
unique_counts_Rehomed <- table(Data$Rehomed) # Get the sum of each unique value in the Rehomed column
print(unique_counts_Rehomed) #99999: 9

unique_Returned <- unique(Data$Returned) # Get unique values in the Returned column
print(unique_Returned) #Yes, No, Unknown, NA
unique_counts_Returned <- table(Data$Returned) # Get the sum of each unique value in the Returned column
print(unique_counts_Returned) #Yes: 23, No: 264, Unknown: 4, NA: 14

# Calculate the number and percentage of rows with missing observations for rehoming time
missing_rehoming_time <- sum(Data$Rehomed == 99999)
missing_rehoming_time_percentage <- (missing_rehoming_time / nrow(Data)) * 100

# Calculate the number and percentage of rows with missing observations for breed
missing_breed <- sum(is.na(Data$Breed))
missing_breed_percentage <- (missing_breed / nrow(Data)) * 100

# Calculate the number and percentage of rows with NA observations for Returned
missing_NA_returned <- sum(is.na(Data$Returned))
missing_NA_returned_percentage <- (missing_NA_returned / nrow(Data)) * 100

# Calculate the number and percentage of rows with Unknown observations for Returned
missing_Unknown_returned <- sum(!is.na(Data$Returned) & Data$Returned == "Unknown")
missing_Unknown_returned_percentage <- (missing_Unknown_returned / nrow(Data)) * 100

# Print the results
print(paste("Number of rows with missing rehoming time: ", missing_rehoming_time))
print(paste("Percentage of rows with missing rehoming time: ", missing_rehoming_time_percentage, "%"))
print(paste("Number of rows with missing breed: ", missing_breed))
print(paste("Percentage of rows with missing breed: ", missing_breed_percentage, "%"))
print(paste("Number of rows with missing NA in Returned: ", missing_NA_returned))
print(paste("Percentage of rows with missing NA in Returned: ", missing_NA_returned_percentage, "%"))
print(paste("Number of rows with missing Unknown in Returned: ", missing_Unknown_returned))
print(paste("Percentage of rows with missing breed: ", missing_Unknown_returned_percentage, "%"))
total_missing_obj_percentage <- (missing_rehoming_time + missing_breed)/nrow(Data) * 100 #We are concerned about the missing values in Rehomed and Breed alone.
print(total_missing_obj_percentage)

#Checking if the the Breed NA and Rehomed 99999 are not in the same rows, so that we know how many percent of rows are we removing
minm <- filter(Data, Rehomed == 99999 | is.na(Breed))
```

```r
print(minm) #No overlapping of NA and 99999 in a same row, thus we remove 15 rows (4.91% of total rows) from the dataset  (305-290)

#Data Cleaning & Flitering
# Remove these rows from the dataset
cleaned_data <- Data %>%
  filter(Rehomed != 99999, !is.na(Breed))

# Print the cleaned data
print(cleaned_data)
print(dim(cleaned_data)) # 290 rows and 7 columns

#----------------------------------------Data Exploration----------------------------------
#Data Exploration:
# Split the data by breed
breeds <- unique(cleaned_data$Breed)
data_by_breed <- split(cleaned_data, cleaned_data$Breed)
print(breeds)
print(data_by_breed)
# Get the number of rows in each type of breed
breed_counts <- table(cleaned_data$Breed)
# Print the number of rows in each type of breed
print(breed_counts)
#Dobermann-21; Labrador Retriever-62; Staffordshire Bull Terrier-207

#Getting Summary
# Create numerical summaries for each dataset
for(breed in breeds) {
  print(paste("Numerical summaries for breed: ", breed))
  print(summary(data_by_breed[[breed]]))
}#Visited Min has negative values. Maybe outliers

# Get unique values for each breed
for(breed in breeds) {
  print(paste("Unique rehoming times for breed: ", breed))
  print(unique(data_by_breed[[breed]]$Rehomed))
}#The value has Discrete data instead of continuous data

# Summarize the rehoming time for each breed
for(breed in breeds) {
  print(paste("Average rehoming time for breed: ", breed))
  print(mean(data_by_breed[[breed]]$Rehomed, na.rm = TRUE))
}
# Create a boxplot of rehoming time for each breed #Figure 1
ggplot(cleaned_data, aes(x = Breed, y = Rehomed,fill = Breed)) +
  geom_boxplot() +
  labs(title = "Boxplot of Rehoming Time by Breed", x = "Breed", y = "Rehoming Time") +
  theme(plot.title = element_text(hjust = 0.5))
```

```r
#----------Data Modelling----------
#Use QQ plot to check if the data is normally distributed for each breed
# Set up a 3-panel plot
par(mfrow = c(1, 3))
par(pty='s')
# Create a QQ plot of rehoming time for each breed #Figure 2
for(breed in breeds) {
  qqnorm(data_by_breed[[breed]]$Rehomed, main = paste(breed), xlab = "Theoretical Quantiles", ylab = "Sample Quantiles", pch = 20, col = "blue")
  qqline(data_by_breed[[breed]]$Rehomed, col = 2, lwd = 2)
}#Right skewed; Light tailed;  Bimodal?

#Use ECDF to check if the data is normally distributed for each breed
# Set up a 3-panel plot
par(mfrow = c(3, 1))
# Create an ECDF of rehoming time for each breed
for(breed in breeds) {
  Fn <- ecdf(data_by_breed[[breed]]$Rehomed)
  plot(Fn, verticals=TRUE ,main = paste("ECDF of Rehoming Time for", breed), xlab = "Rehoming Time", ylab = "ECDF")
  # Add a curve
  curve(pnorm(x, mean = mean(data_by_breed[[breed]]$Rehomed, na.rm = TRUE), sd = sd(data_by_breed[[breed]]$Rehomed, na.rm = TRUE)), add = TRUE, col = "red", lwd = 2)
} #We won't use this, as this does not give perfect results for all breeds

#Use Histogram to check if the data is normally distributed for each breed #Figure 3
# Set up a 3-panel plot
par(mfrow = c(1, 3))
# Create a histogram of rehoming time for each breed
breed_colors <- c("skyblue", "lightpink", "lightgreen")  # Define colors for each breed
i <- 1  # Initialize counter
for(breed in breeds) {
  hist(data_by_breed[[breed]]$Rehomed, main = paste(breed), xlab = "Rehoming Time", ylab = "Count", freq = FALSE, col = breed_colors[i], xlim = c(0, 60), ylim = c(0, 0.06))
  lines(density(data_by_breed[[breed]]$Rehomed, na.rm = TRUE), col = "#4B0082")
  i <- i + 1  # Increment counter
}
#----------Findings & Insights----------
#Checking Distribution for Normality:
mu_dobermann <- mean(data_by_breed[["Dobermann"]]$Rehomed, na.rm = TRUE)
sigma_dobermann <- sd(data_by_breed[["Dobermann"]]$Rehomed, na.rm = TRUE)
mu_labrador <- mean(data_by_breed[["Labrador Retriever"]]$Rehomed, na.rm = TRUE)
sigma_labrador <- sd(data_by_breed[["Labrador Retriever"]]$Rehomed, na.rm = TRUE)
mu_staffordshire <- mean(data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, na.rm = TRUE)
sigma_staffordshire <- sd(data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, na.rm = TRUE)
#KS Test for Normality
```

```
ks.test(x = data_by_breed[["Dobermann"]]$Rehomed, y = "pnorm", mean = mu_dobermann, sd = sigma_dobermann)
ks.test(x = data_by_breed[["Labrador Retriever"]]$Rehomed, y = "pnorm", mean = mu_labrador, sd = sigma_labrador)
ks.test(x = data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, y = "pnorm", mean = mu_staffordshire, sd = sigma_staffordshire)

#Shapiro-Wilk Test for Normality
shapiro.test(data_by_breed[["Dobermann"]]$Rehomed)
shapiro.test(data_by_breed[["Labrador Retriever"]]$Rehomed)
shapiro.test(data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed)

#Chi-Square goodness of Fit Test for Normality
pearson.test(data_by_breed[["Dobermann"]]$Rehomed)
pearson.test(data_by_breed[["Labrador Retriever"]]$Rehomed)
pearson.test(data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed) #We will not use this, as our data is not categorical

#Average Rehoming Time:
# Calculate a confidence interval for the mean rehoming time for each breed
par(mfrow = c(1, 1))
#T test
ci_dobermann_t <- t.test(data_by_breed[["Dobermann"]]$Rehomed, mu = 27)$conf.int

#Z test
ci_labrador_z <- z.test(data_by_breed[["Labrador Retriever"]]$Rehomed, mu = 27, sigma.x = sqrt(74))$conf.int
ci_staffordshire_z <- z.test(data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, mu = 27, sigma.x = sqrt(74))$conf.int
# Create a data frame to summarize the results
ci_summary <- data.frame(
  Breed = c("Dobermann", "Labrador Retriever", "Staffordshire Bull Terrier"),
  Lower_Bound = c(ci_dobermann_t[1], ci_labrador_z[1], ci_staffordshire_z[1]), #Since we choose t test for dobermann and z test for the other two.
  Upper_Bound = c(ci_dobermann_t[2], ci_labrador_z[2], ci_staffordshire_z[2])
)
# Print the summary
print(ci_summary)

# Plotting the confidence intervals
par(pty='m')
# Analysis labels for the left side:
analysis = c("Dobermann",
"Labrador Retriever",
"Staffordshire Bull Terrier")
# Results of each test (estimated mean, upper CI limit, lower CI limit, p-value):
estimate = c(17.85714, 19.85484, 19.33816)
upper = c(22.65569, 21.99609, 20.51003)
lower = c(13.05859, 17.71359, 18.16630)
```

```
pval = c("7.46e-4","6.143e-11","2.2e-16")
# Note that the order of the results in each vector must match the order of the labels in the vector "analysis". Set the margin widths:
par(mar = c(5,6,0,8))
# Create an empty plot of a suitable size (considering the width of your confidence intervals):
plot(x = 0, # One point at (0,0).
xlim = c(-5, 30), ylim=c(0.75, 3), # Axis limits.
type = "n", xaxt = "n", yaxt="n", # No points, no axes drawn.
xlab = NULL, ylab= NULL, ann = FALSE, # No axis labels or numbers.
bty="n") # No box.
# Add a horizontal (side = 1) axis:
axis(side = 1, cex.axis = 1)
# Add an axis label 4 lines below the axis:
mtext("Rehoming Time",
side = 1, line = 2)
# Add some grid lines, preferably lined up with the numbers on the horizontal axis:
for(i in c(0, 10, 20)){
lines(c(i, i), c(0, 2.25), lty = 2, col = "gray53")
}
# Add labels for each analysis on the left (side = 2) at vertical heights of 1, 2, and 3:
verticalpos = seq(1, 2, length.out = length(analysis)) # Adjust the gap here
mtext(text = analysis, at = verticalpos,
side = 2, line = 5, outer = FALSE, las = 1, adj = 0)
# Try changing the "line" option to move these closer to or away from the plotted intervals.
# Plot the four point estimates (centres of the CIs for each analysis):
points(estimate, verticalpos, pch = 16)
# Plot the three interval estimates:
for(i in 1:3 ){
lines(c(lower[i], upper[i]), c(verticalpos[i], verticalpos[i]))
lines(c(lower[i], lower[i]), c(verticalpos[i] + 0.2, verticalpos[i] - 0.2))
lines(c(upper[i], upper[i]), c(verticalpos[i] + 0.2, verticalpos[i] - 0.2))}
# Now we add numerical results on the right (side = 4), but we need to put them into a nice form first. Note that paste() merges words and numbers, and formatC() allows us to control the number of decimal places.
est <- formatC(estimate, format='f', digits = 2)
P <- formatC(pval , format = 'f', digits = 7)
pval <- paste("p =", P) # Type pval to see what this does.
L <- formatC(lower, format = 'f', digits = 2)
U <- formatC(upper, format = 'f', digits = 2)
interval <- paste("(", L, ", ", U, "),", sep = "") # Type interval to check.
# Putting it all together:
results <- paste(est, interval, pval)
# Add to the plot:
mtext(text = results, at = verticalpos,
side = 4, line = 4, outer = FALSE, las = 1, adj = 1)
# Like a Christmas present, an R plot belongs in a box:
box("inner")
```

```r
# Add a title to the plot
title("Confidence Intervals for Mean Rehoming Time",line=-10)


#Comparison of Breeds:
# Calculate a confidence interval for the difference in mean rehoming times between each pair of breeds
ci_dobermann_labrador <- t.test(data_by_breed[["Dobermann"]]$Rehomed, data_by_breed[["Labrador Retriever"]]$Rehomed, var.equal = TRUE)$conf.int
ci_dobermann_staffordshire <- t.test(data_by_breed[["Dobermann"]]$Rehomed, data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, var.equal = TRUE)$conf.int
ci_labrador_staffordshire <- t.test(data_by_breed[["Labrador Retriever"]]$Rehomed, data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, var.equal = TRUE)$conf.int
# Create a data frame to summarize the results
ci_diff_summary <- data.frame(
  Pair = c("Dobermann vs Labrador Retriever", "Dobermann vs Staffordshire Bull Terrier", "Labrador Retriever vs Staffordshire Bull Terrier"),
  Lower_Bound = c(ci_dobermann_labrador[1], ci_dobermann_staffordshire[1], ci_labrador_staffordshire[1]),
  Upper_Bound = c(ci_dobermann_labrador[2], ci_dobermann_staffordshire[2], ci_labrador_staffordshire[2])
)
print(ci_diff_summary) # Print the summary

# Calculate the p-value for the difference in mean rehoming times between each pair of breeds
p_dobermann_labrador <- t.test(data_by_breed[["Dobermann"]]$Rehomed, data_by_breed[["Labrador Retriever"]]$Rehomed, var.equal = TRUE)$p.value
p_dobermann_staffordshire <- t.test(data_by_breed[["Dobermann"]]$Rehomed, data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, var.equal = TRUE)$p.value
p_labrador_staffordshire <- t.test(data_by_breed[["Labrador Retriever"]]$Rehomed, data_by_breed[["Staffordshire Bull Terrier"]]$Rehomed, var.equal = TRUE)$p.value
# Create a data frame to summarize the results
p_value_summary <- data.frame(
  Pair = c("Dobermann vs Labrador Retriever", "Dobermann vs Staffordshire Bull Terrier", "Labrador Retriever vs Staffordshire Bull Terrier"),
  P_Value = c(p_dobermann_labrador, p_dobermann_staffordshire, p_labrador_staffordshire) )
print(p_value_summary) # Print the summary
#-----------------------------------------Discussion-----------------------------------------
#Transformation of Rehoming Time to Logarithmic Scale
# Create a new column for the log of rehoming time
cleaned_data$log_rehoming_time <- log(cleaned_data$Rehomed)
# Create a boxplot of the log of rehoming time for each breed #Figure 5
ggplot(cleaned_data, aes(x = Breed, y = log_rehoming_time,fill = Breed)) +
  geom_boxplot() +
  labs(title = "Boxplot of Log Rehoming Time by Breed", x = "Breed", y = "Log Rehoming Time") +
  theme(plot.title = element_text(hjust = 0.5))
```

```r
# Split the data by breed
breeds <- unique(cleaned_data$Breed)
data_by_breed <- split(cleaned_data, cleaned_data$Breed)
print(breeds)
print(data_by_breed)
# Get the number of rows in each type of breed
breed_counts <- table(cleaned_data$Breed)
# Print the number of rows in each type of breed
print(breed_counts)

#Use QQ plot to check if the data is normally distributed for each breed
# Set up a 3-panel plot
par(mfrow = c(1, 3))
par(pty='s')
# Create a QQ plot of rehoming time for each breed #Figure 6
for(breed in breeds) {
 qqnorm(data_by_breed[[breed]]$log_rehoming_time, main = paste(breed), xlab =
"Theoretical Quantiles", ylab = "Sample Quantiles", pch = 20, col = "blue")
 qqline(data_by_breed[[breed]]$log_rehoming_time, col = 2, lwd = 2)
}

#Future Work using Weibull Distribution
library(MASS)
#Not Exponential, neither Poisson, It is skewed
par(pty='m',mfrow = c(1, 1))
fit_weibull <- fitdistr(data_by_breed[["Dobermann"]]$Rehomed,"weibull")
hist(data_by_breed[["Dobermann"]]$Rehomed, breaks = "Sturges", probability = TRUE,
    main = "Histogram of Rehoming Times with Fitted Weibull Curve",
    xlab = "Rehoming Times", ylab = "Density")
# Function to generate Weibull density values
weibull_density <- function(x, shape, scale) {
 dweibull(x, shape = shape, scale = scale)
}
# Extract the shape and scale parameters from the fit
shape_param <- fit_weibull$estimate["shape"]
scale_param <- fit_weibull$estimate["scale"]
# Add the curve using the shape and scale parameters obtained from the fit
curve(weibull_density(x, shape_param, scale_param), add = TRUE, col = "red", lwd = 2,
from=0, to=max(data_by_breed[["Dobermann"]]$Rehomed))
```