Untitled

Gerard Corrales

2024-11-17

```r
# Libraries
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(ggplot2)
```

```r
# Importing dataset
intakes.data <- read.csv("Austin_Animal_Center_Intakes.csv")
```

EDA

```r
# Checking structure of the dataset
str(intakes.data)
```

```
## 'data.frame':    168998 obs. of  12 variables:
##  $ Animal.ID       : chr  "A786884" "A706918" "A724273" "A665644" ...
##  $ Name            : chr  "*Brock" "Belle" "Runster" "" ...
##  $ DateTime        : chr  "01/03/2019 04:19:00 PM" "07/05/2015 12:59:00 PM" "04/14/2016 06:43:00 PM"
##  $ MonthYear       : chr  "January 2019" "July 2015" "April 2016" "October 2013" ...
##  $ Found.Location  : chr  "2501 Magin Meadow Dr in Austin (TX)" "9409 Bluegrass Dr in Austin (TX)" "2
##  $ Intake.Type     : chr  "Stray" "Stray" "Stray" "Stray" ...
##  $ Intake.Condition: chr  "Normal" "Normal" "Normal" "Sick" ...
##  $ Animal.Type     : chr  "Dog" "Dog" "Dog" "Cat" ...
```

```
##  $ Sex.upon.Intake : chr  "Neutered Male" "Spayed Female" "Intact Male" "Intact Female" ...
##  $ Age.upon.Intake : chr  "2 years" "8 years" "11 months" "4 weeks" ...
##  $ Breed           : chr  "Beagle Mix" "English Springer Spaniel" "Basenji Mix" "Domestic Shorthair 
##  $ Color           : chr  "Tricolor" "White/Liver" "Sable/White" "Calico" ...
```

```r
# View first rows
head(intakes.data)
```

```
##   Animal.ID         Name               DateTime    MonthYear
## 1   A786884       *Brock 01/03/2019 04:19:00 PM January 2019
## 2   A706918        Belle 07/05/2015 12:59:00 PM    July 2015
## 3   A724273      Runster 04/14/2016 06:43:00 PM   April 2016
## 4   A665644              10/21/2013 07:59:00 AM October 2013
## 5   A857105 Johnny Ringo 05/12/2022 12:23:00 AM     May 2022
## 6   A682524          Rio 06/29/2014 10:38:00 AM    June 2014
##                         Found.Location   Intake.Type Intake.Condition
## 1 2501 Magin Meadow Dr in Austin (TX)         Stray           Normal
## 2    9409 Bluegrass Dr in Austin (TX)         Stray           Normal
## 3  2818 Palomino Trail in Austin (TX)         Stray           Normal
## 4                         Austin (TX)         Stray             Sick
## 5  4404 Sarasota Drive in Austin (TX) Public Assist           Normal
## 6       800 Grove Blvd in Austin (TX)         Stray           Normal
##   Animal.Type Sex.upon.Intake Age.upon.Intake
## 1         Dog   Neutered Male         2 years
## 2         Dog   Spayed Female         8 years
## 3         Dog     Intact Male       11 months
## 4         Cat   Intact Female         4 weeks
## 5         Cat   Neutered Male         2 years
## 6         Dog   Neutered Male         4 years
##                                    Breed        Color
## 1                             Beagle Mix     Tricolor
## 2               English Springer Spaniel  White/Liver
## 3                            Basenji Mix  Sable/White
## 4                  Domestic Shorthair Mix       Calico
## 5                     Domestic Shorthair Orange Tabby
## 6 Doberman Pinsch/Australian Cattle Dog     Tan/Gray
```

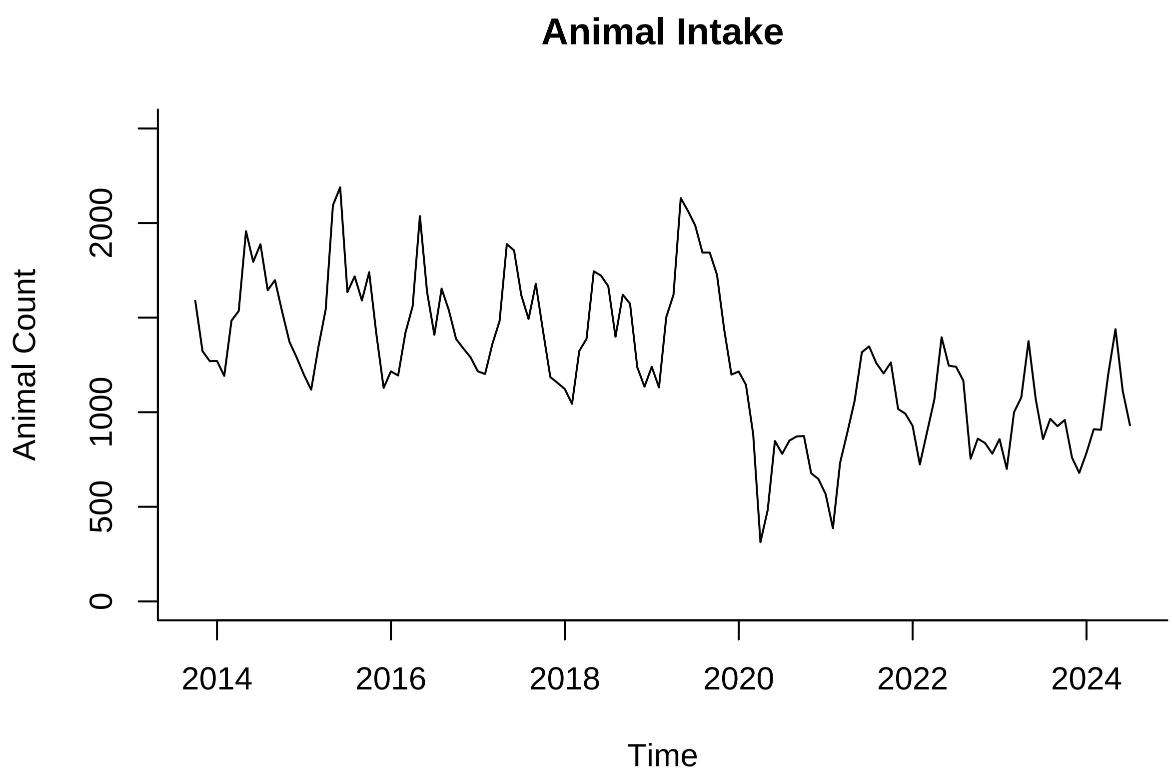
```r
# Checking missing values
colSums(is.na(intakes.data))
```

```
##        Animal.ID             Name         DateTime        MonthYear 
##                0                0                0                0 
##   Found.Location      Intake.Type Intake.Condition      Animal.Type 
##                0                0                0                0 
##  Sex.upon.Intake  Age.upon.Intake            Breed            Color 
##                0                0                0                0
```

We don't have any missing values in any column.

```r
# Getting the correct Date format
intakes.data$MonthYear <- as.Date(intakes.data$DateTime, format = "%m/%d/%Y %H:%M")

# Count how many animals are in each month and put it in a new column 'month'
monthly_counts <- intakes.data %>%
  mutate(month = format(MonthYear, "%Y-%m")) %>%
  group_by(month) %>%
  summarise(count = n())
```

```r
# Convert data frame in Time Series object
month.ts <- ts(monthly_counts$count, start = c(2013, 10), end = c(2024, 7), frequency = 12)

plot(month.ts, xlab = "Time", ylab = "Animal Count", ylim = c(0, 2500), bty = "l", main= "Animal Intake
```



This graph shows an exceptional decline in animal intakes during 2020. Also we can see a strong seasonality through the entire graph where during the same months there is a peak and then a decline. And two different trends, from 2013 until 2020 the trend seems slightly declining but quite steady and after the big trough in 2020 the trend starts increasing.

```r
# Summary statistics
# Total number of observations
total_observations <- nrow(intakes.data)

# Timeframe of the dataset
timeframe <- range(intakes.data$MonthYear, na.rm = TRUE)

# Summary table: Count of animals by type
animal_type_summary <- intakes.data %>%
  group_by(Animal.Type) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

# Print summary statistics
print(paste("Total observations:", total_observations))
```

```
## [1] "Total observations: 168998"
```

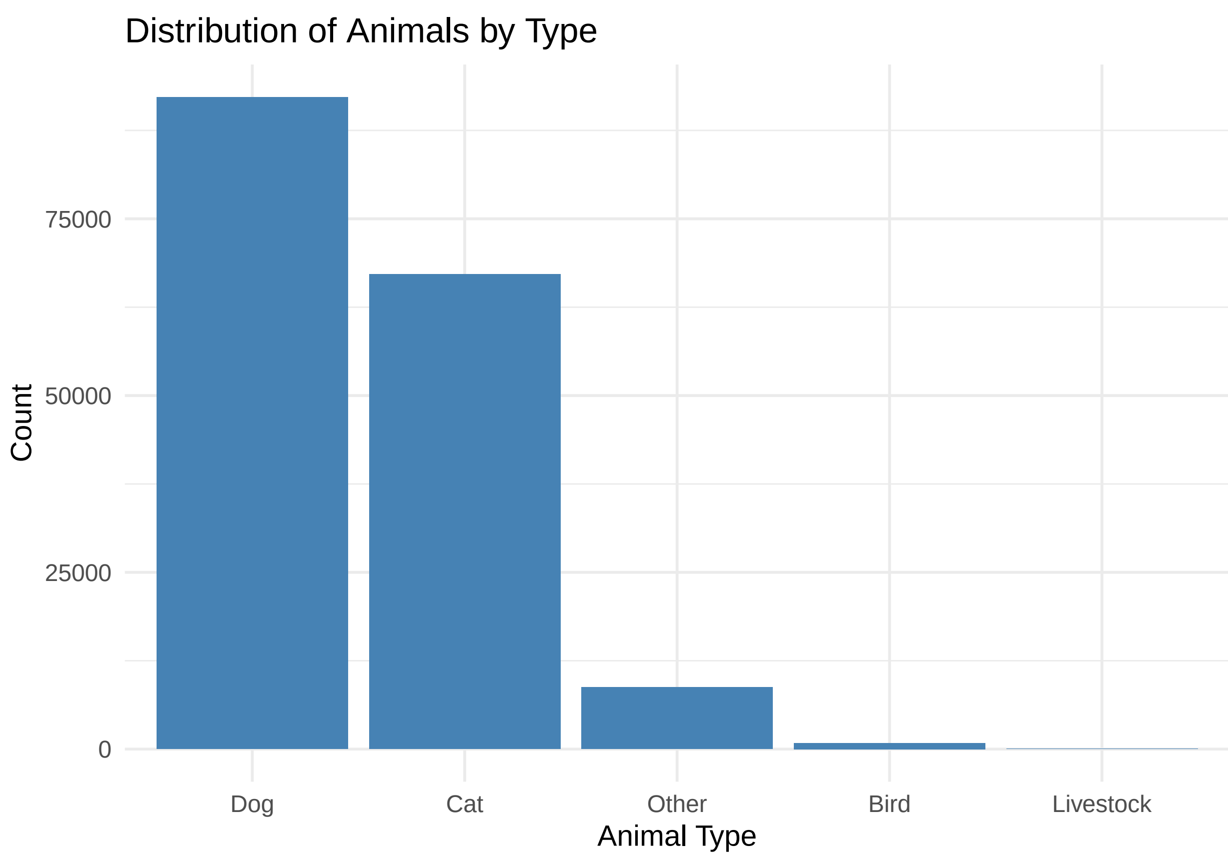
```r
print(paste("Timeframe:", format(timeframe[1], "%Y-%m-%d"), "to", format(timeframe[2], "%Y-%m-%d")))
```

```
## [1] "Timeframe: 2013-10-01 to 2024-11-19"
```

```r
# Print summary statistics
print(animal_type_summary)
```

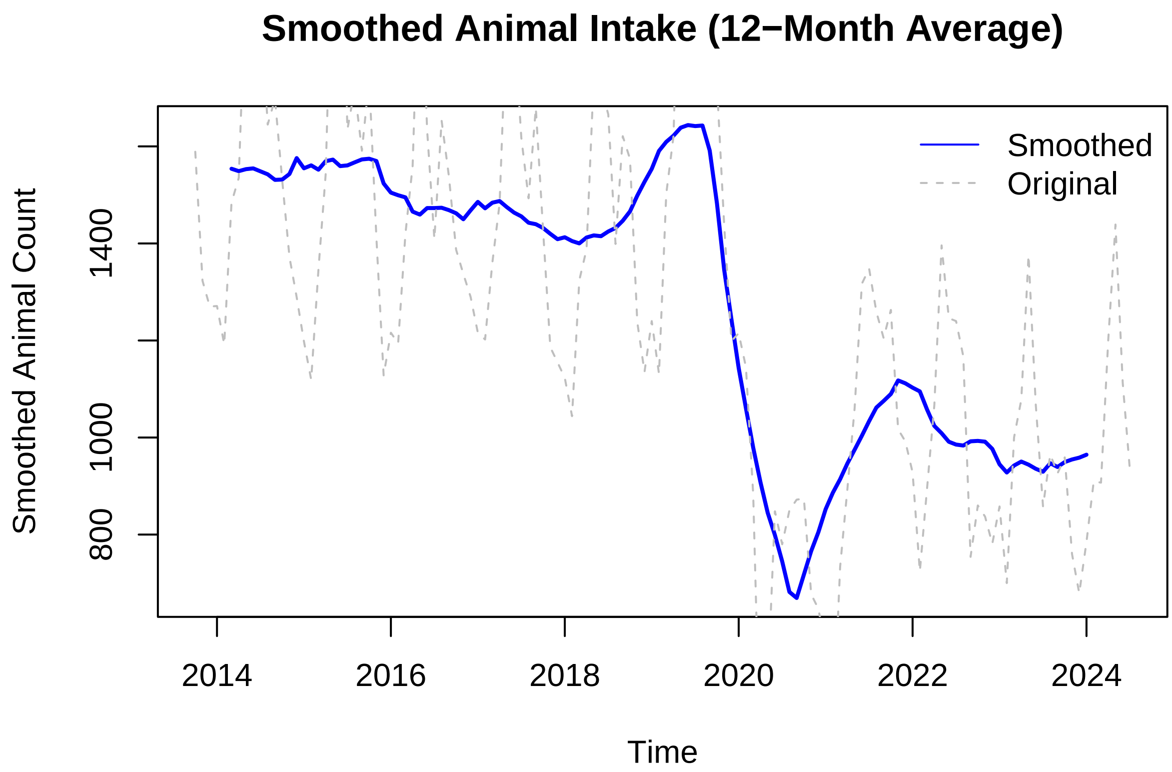
```
## # A tibble: 5 x 2
##   Animal.Type Count
##   <chr>       <int>
## 1 Dog         92204
## 2 Cat         67160
## 3 Other        8747
## 4 Bird          854
## 5 Livestock      33
```

```r
# Visualization: Bar chart showing distribution of animals by type
ggplot(animal_type_summary, aes(x = reorder(Animal.Type, -Count), y = Count)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  theme_minimal() +
  labs(
    title = "Distribution of Animals by Type",
    x = "Animal Type",
    y = "Count"
  )
```

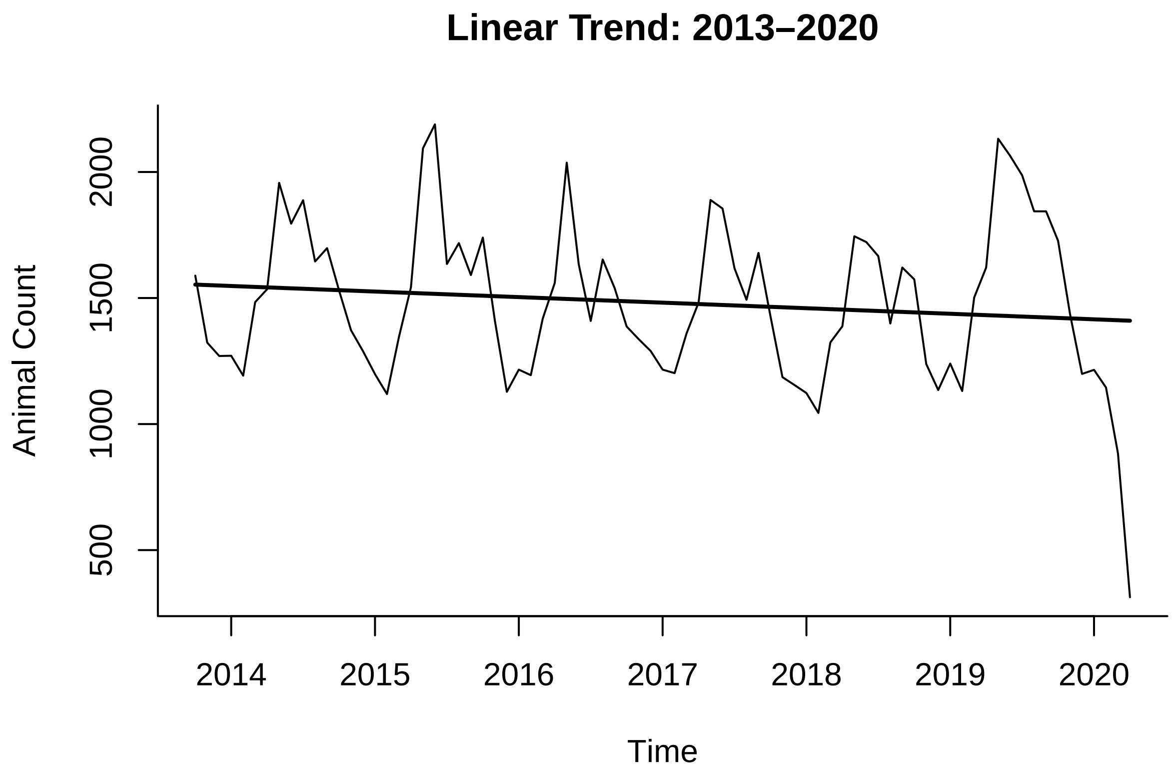



```r
# Apply a 12-month moving average for smoothing
smooth.ts <- rollmean(month.ts, k = 12, align = "center", fill = NA)
```

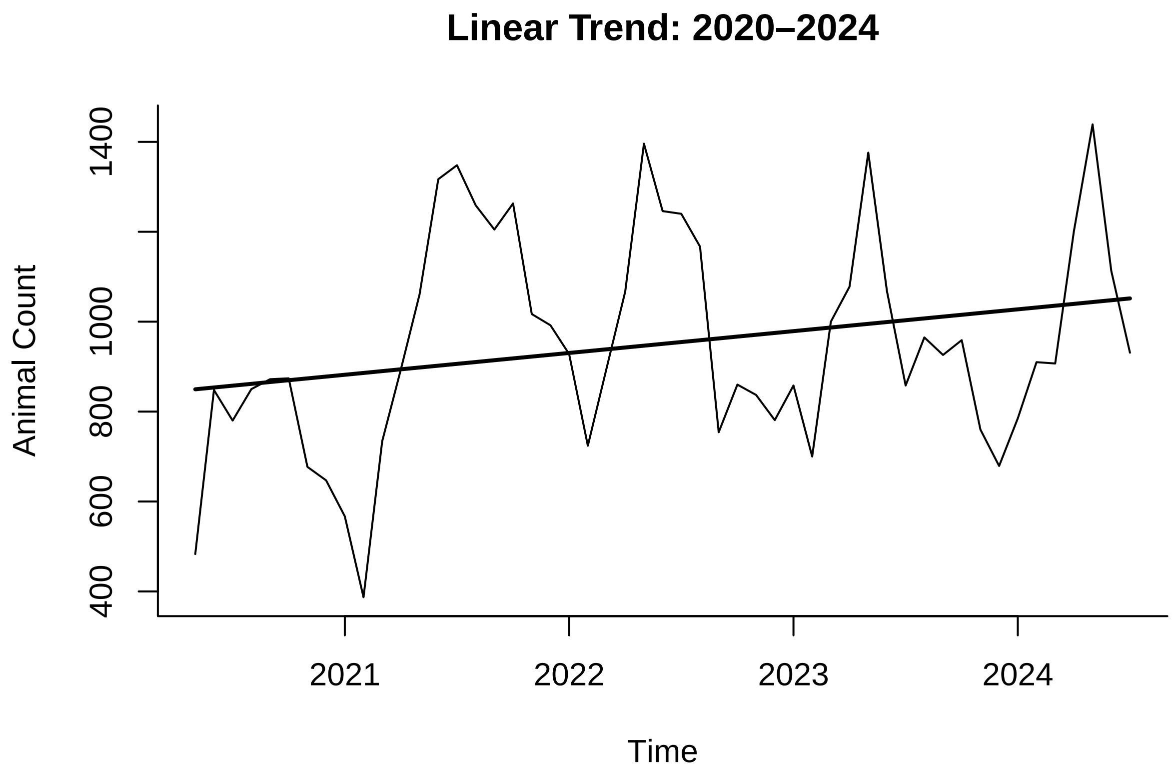
```r
# Plot the smoothed time series
plot(smooth.ts, xlab = "Time", ylab = "Smoothed Animal Count",
     main = "Smoothed Animal Intake (12-Month Average)", col = "blue", lwd = 2)

# Optional: Overlay the original time series for comparison
lines(month.ts, col = "gray", lty = 2)
legend("topright", legend = c("Smoothed", "Original"), col = c("blue", "gray"), lty = c(1, 2), bty = "n
```




```r
# Original splitted series with zoom in to better understand trends
# Creating two subsets to zoom in 2013-2020 and 2020-2024
month.ts.2013_2020 <- window(month.ts, start = c(2013, 10), end = c(2020, 4))
month.ts.2020_2024 <- window(month.ts, start = c(2020, 5), end = c(2024, 7))

# Fitting separate straight-line trends
trend_2013_2020 <- tslm(month.ts.2013_2020 ~ trend)
trend_2020_2024 <- tslm(month.ts.2020_2024 ~ trend)

# Plot for 2013-2020 with linear trend line
plot(month.ts.2013_2020, xlab = "Time", ylab = "Animal Count", bty = "l", main = "Linear Trend: 2013-20
lines(trend_2013_2020$fitted, lwd = 2)
```




```
# Plot for 2020-2024 with linear trend line
plot(month.ts.2020_2024, xlab = "Time", ylab = "Animal Count", bty = "l", main = "Linear Trend: 2020-202
lines(trend_2020_2024$fitted, lwd = 2)
```




```r
cat("Trend: 2013 to 2020")
```

```
## Trend: 2013 to 2020
```

```r
cat("\n")
```

```r
summary(trend_2013_2020)$coefficients
```

```
##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 1555.101266  71.698484 21.689458 1.561237e-34
## trend         -1.836392   1.557192 -1.179298 2.419107e-01
```

```r
cat("\n")
```

```r
cat("Trend: 2020 to 2024")
```

```
## Trend: 2020 to 2024
```

```r
cat("\n")
```

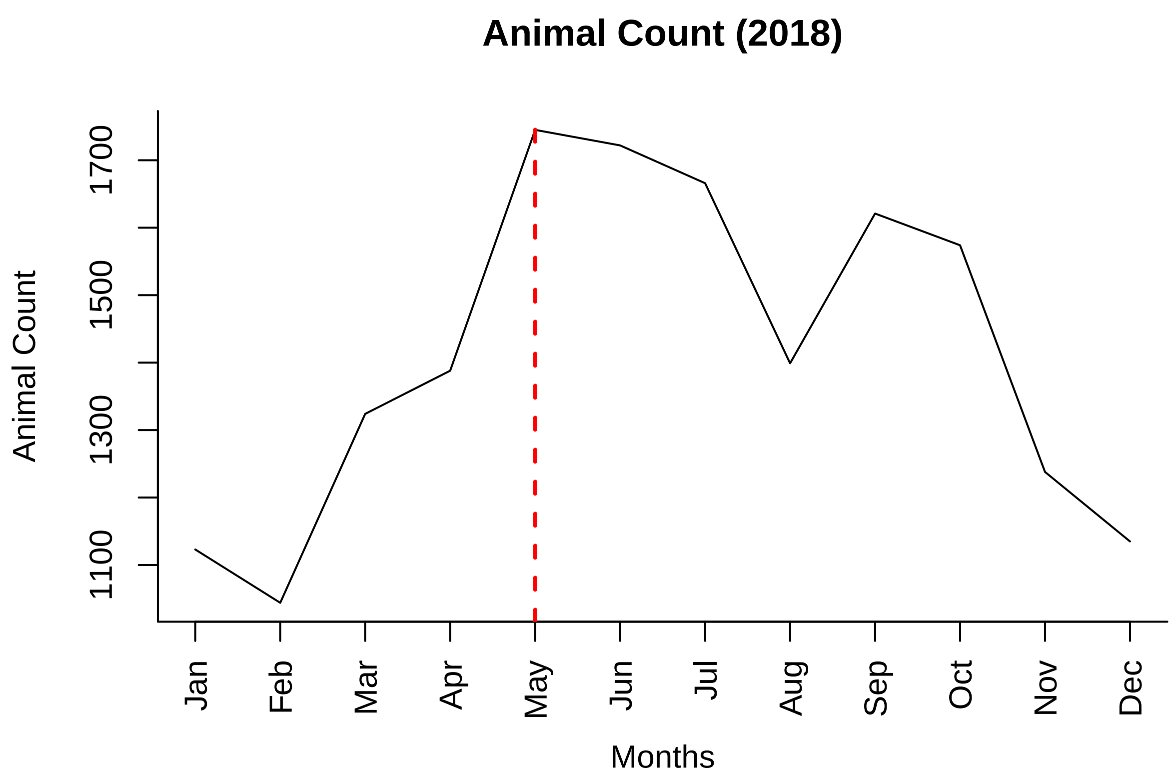
```r
summary(trend_2020_2024)$coefficients
```

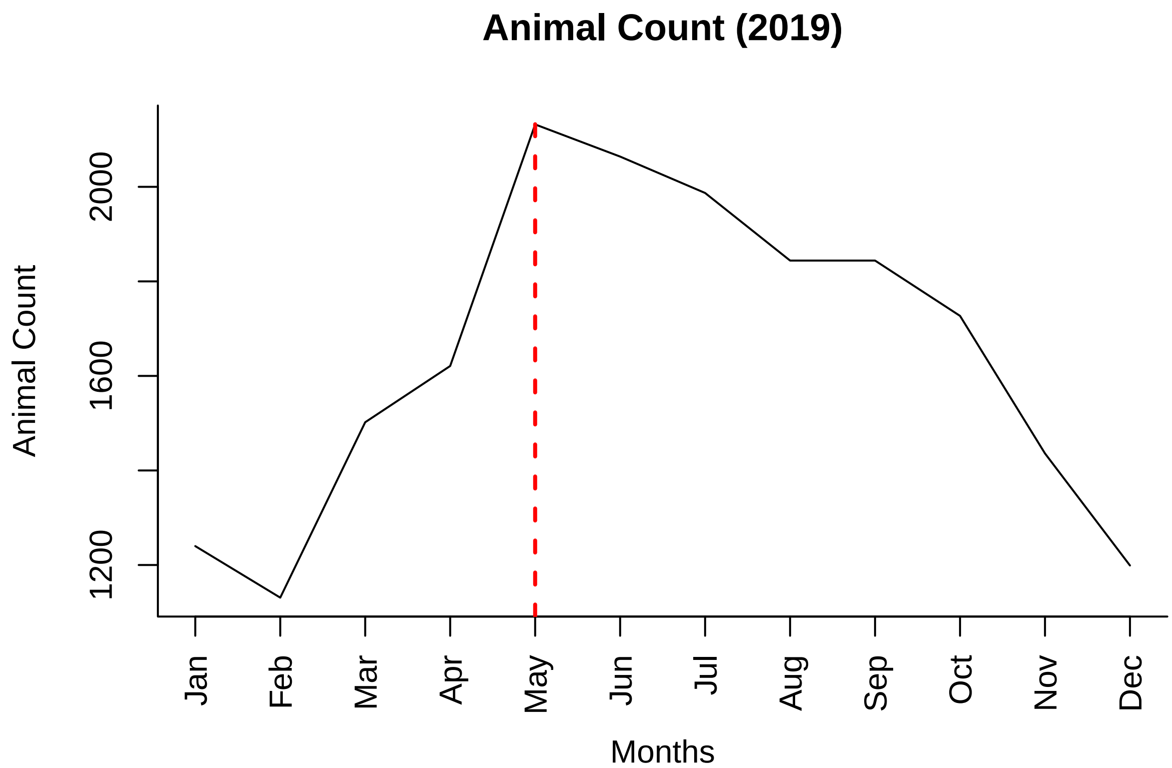
```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 845.539608  66.546714 12.705956 4.006176e-17
## trend         4.043348   2.227317  1.815345 7.559319e-02
```

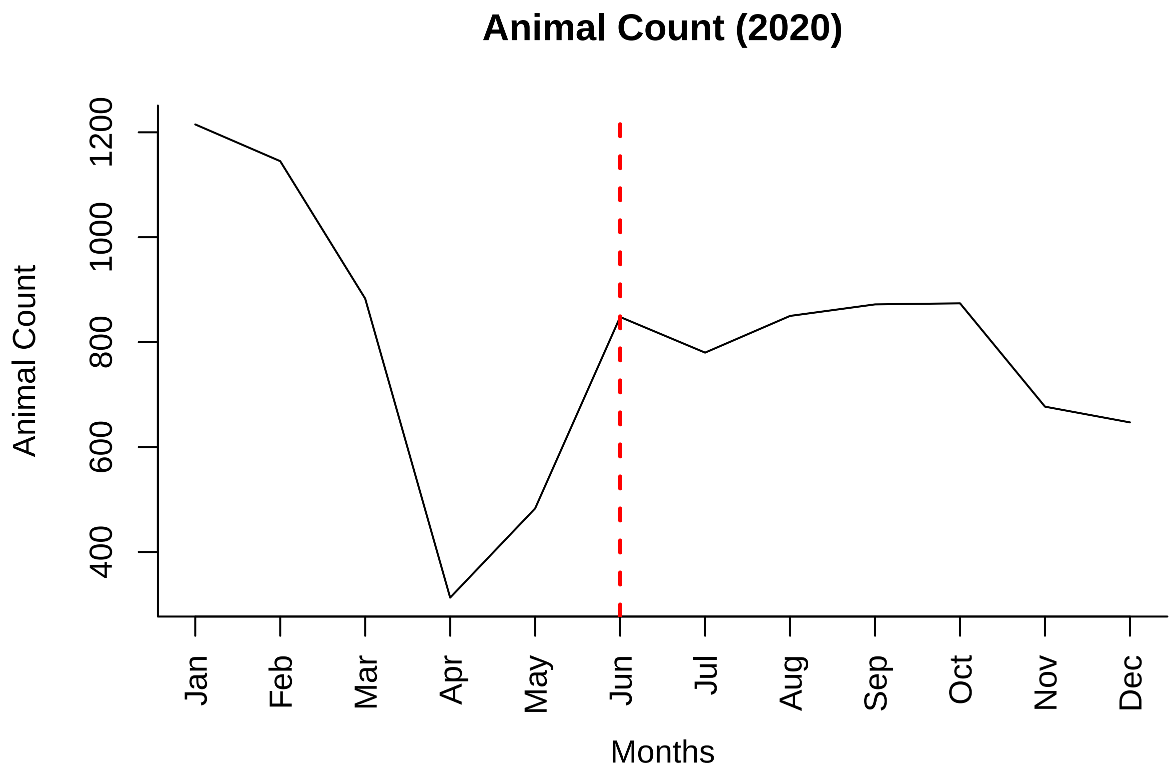
The first trend (2013-2020) has a slight negative slope indicating some decline over those years. While in the second trend (2020-2024) has a more positive slope indicating higher volume in animal intakes.

```r
# Which months receive the most animals?
# Zooming in for each full year to discover which months have peaks or troughs
```

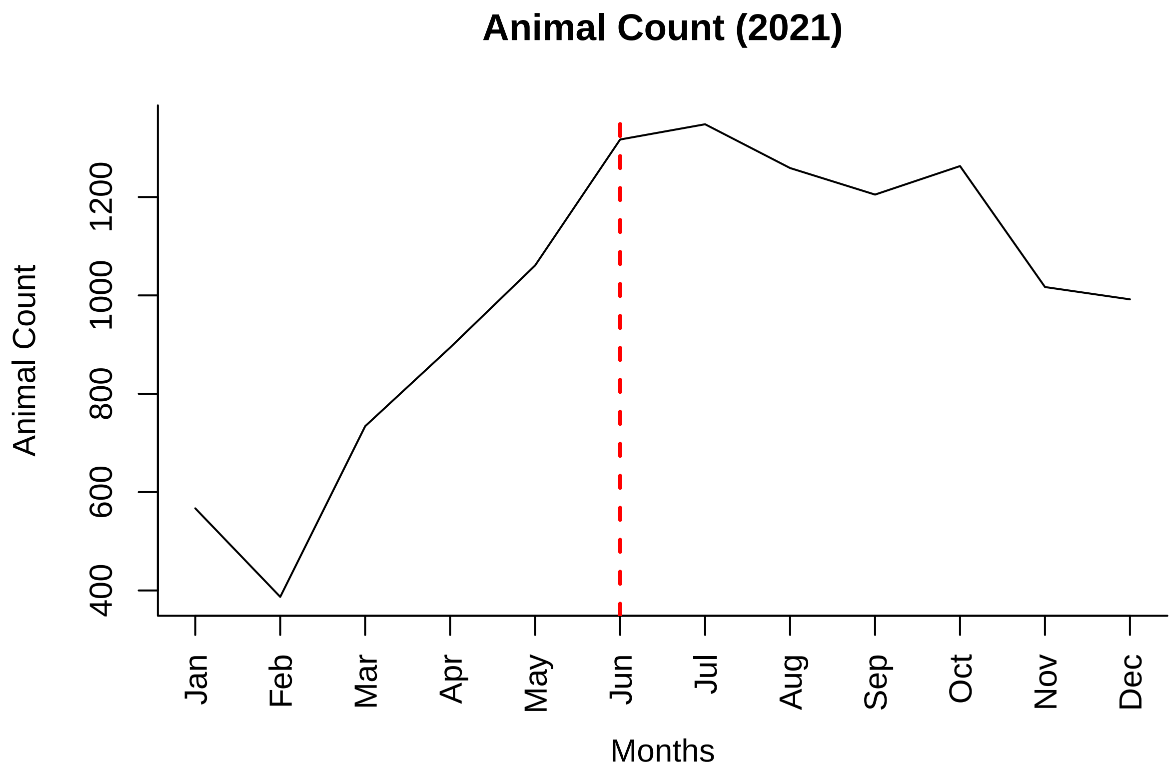
```r
years <- 2018:2023

for (year in years) {

  year_data <- window(month.ts, start = c(year, 1), end = c(year, 12))

  # Plot
  plot(year_data,
       xaxt = "n",
       xlab = "Months",
       ylab = "Animal Count",
       main = paste("Animal Count (", year, ")", sep = ""), bty = "l")

    axis(1, at = time(year_data), labels = month.abb, las = 2)

  if (year == 2020 || year == 2021) {

    abline(v = time(year_data)[6], col = "red", lwd = 2, lty = 2)
  } else {

    abline(v = time(year_data)[5], col = "red", lwd = 2, lty = 2)
  }

}
```

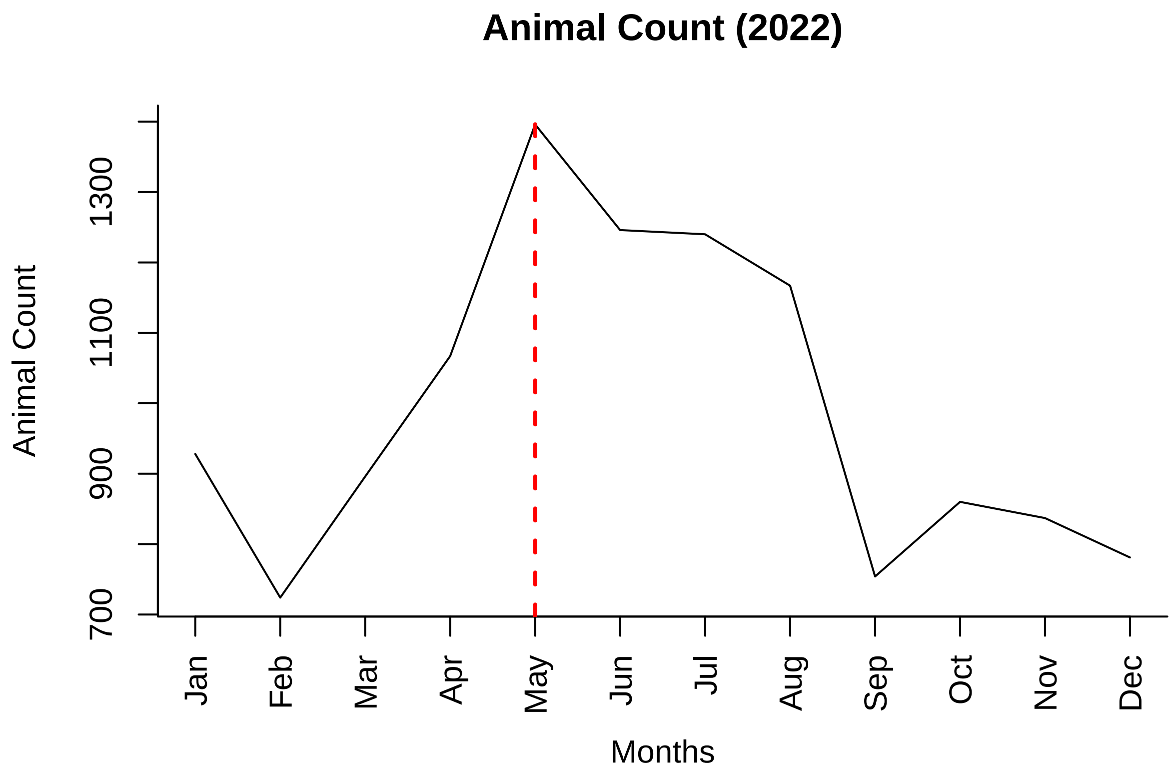


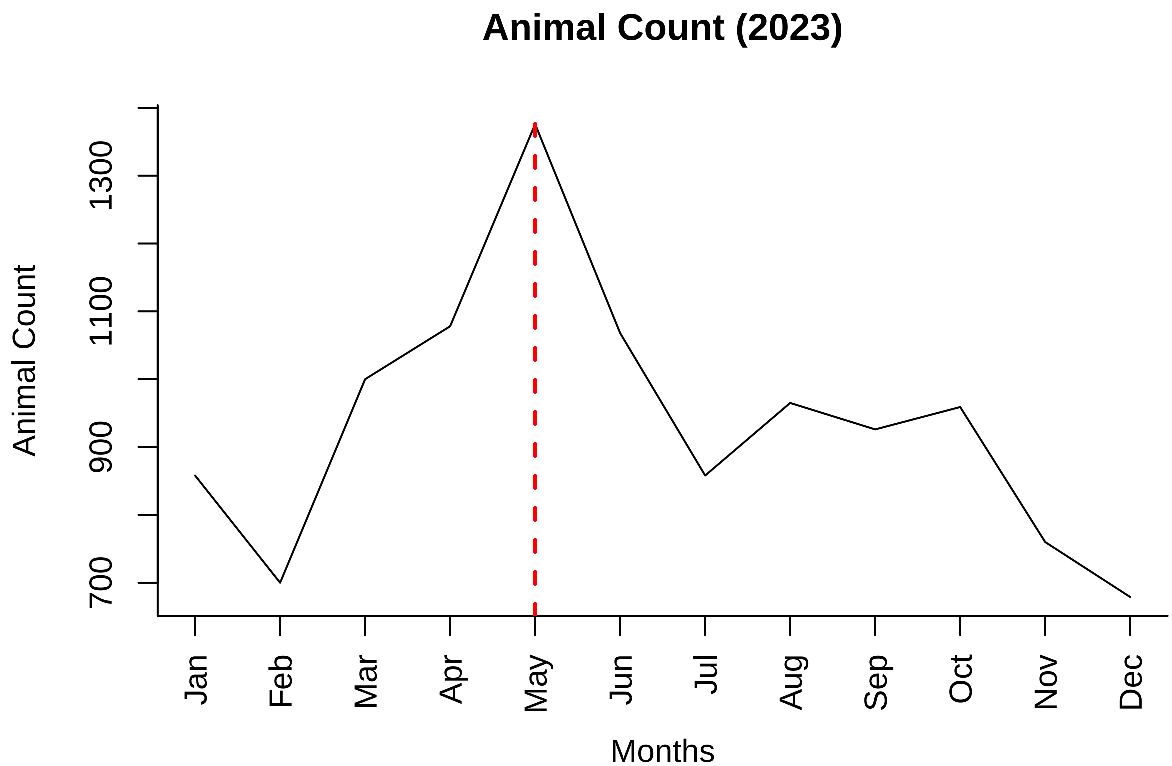

Animal Count (2019)
Animal Count
2000
1600
1200
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Months



Animal Count (2020)
Animal Count
400
600
800
1000
1200
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Months


Animal Count (2021)
Animal Count
400
600
800
1000
1200
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Months


Animal Count (2022)
Animal Count
700
900
1100
1300
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Months

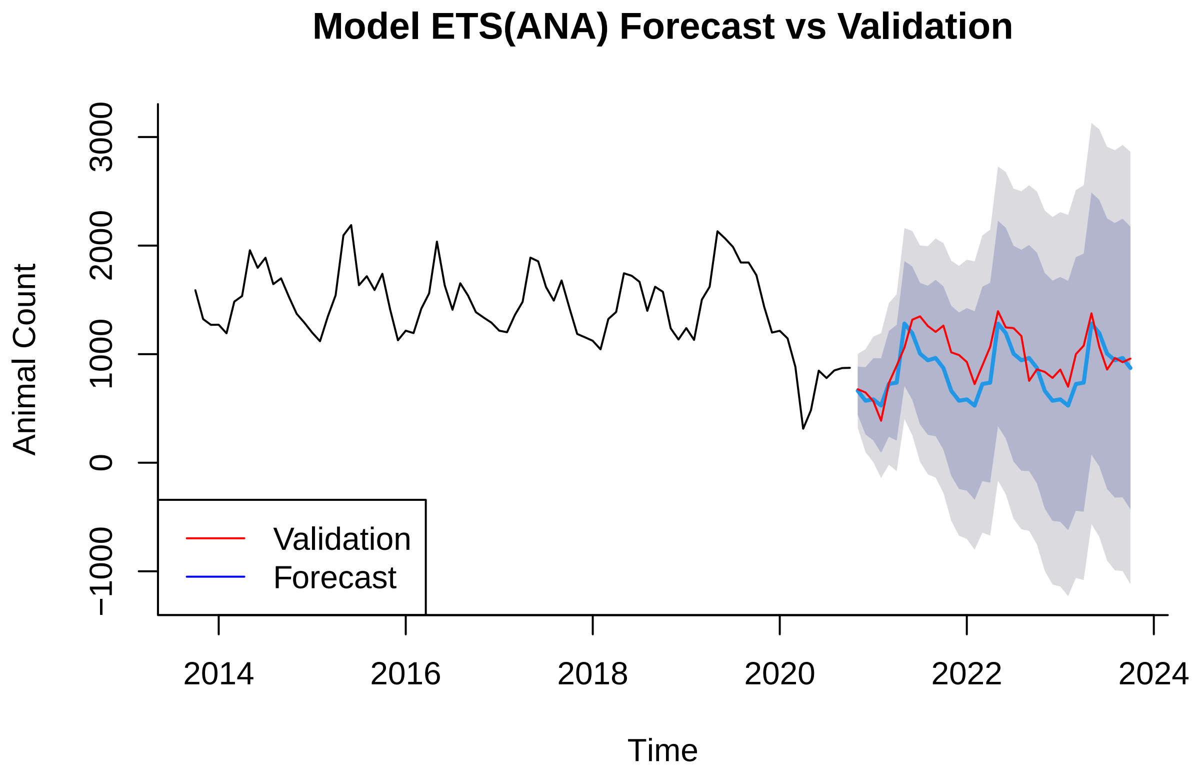


The graphs consistently show (seasonality) peaks during the month of May for most years (2014, 2016–2019, 2022, and 2023), with exceptions in June (2015, 2020) and June–July (2021). Additionally, all graphs display another noticeable increase during October each year.

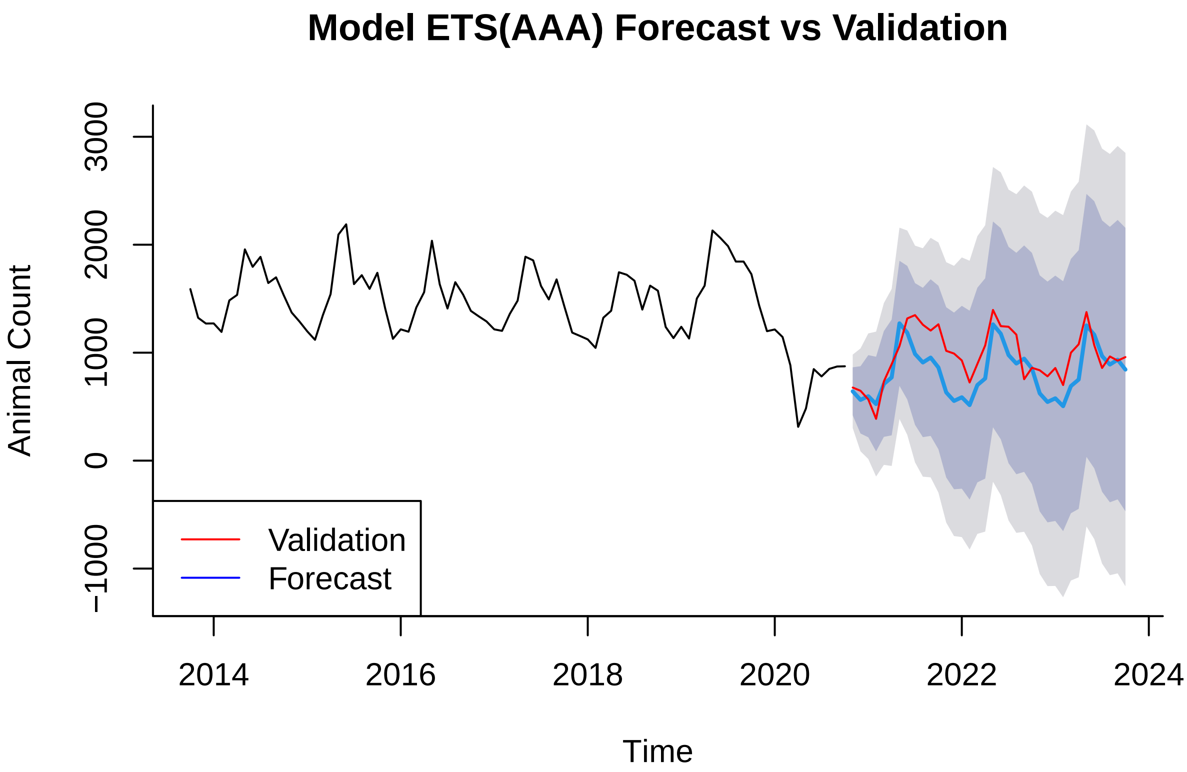
Model Selection

```
# Performance Evaluation

# Number of observations during the validation period (months, in this case)
nValid <- 36

# Calculating number of training observations
nTrain <- length(month.ts) - nValid

# Splitting data into training and validation periods
train.ts <- window(month.ts, start = c(2013, 10), end = c(2013, nTrain))
valid.ts <- window(month.ts, start = c(2013, nTrain + 1), end = c(2013, nTrain + nValid))

# Fit ETS model on training data
train.ets.ANA <- ets(train.ts, model = "ANA") # Additive level, no trend, additive seasonality
train.ets.AAA <- ets(train.ts, model = "AAA") # Additive level, additive trend, additive seasonality
train.ets.MNM <- ets(train.ts, model = "MNM") # Multiplicative level, no trend, multiplicative seasonal
train.ets.ANN <- train.ets.MNM <- ets(train.ts, model = "ANN") # Additive level, no trend, no seasonali
sarima_model <- auto.arima(train.ts, seasonal = TRUE, stepwise = FALSE, approximation = FALSE)

# Forecast for the validation period
train.ets.ANA.pred <- forecast(train.ets.ANA, h = nValid)
```

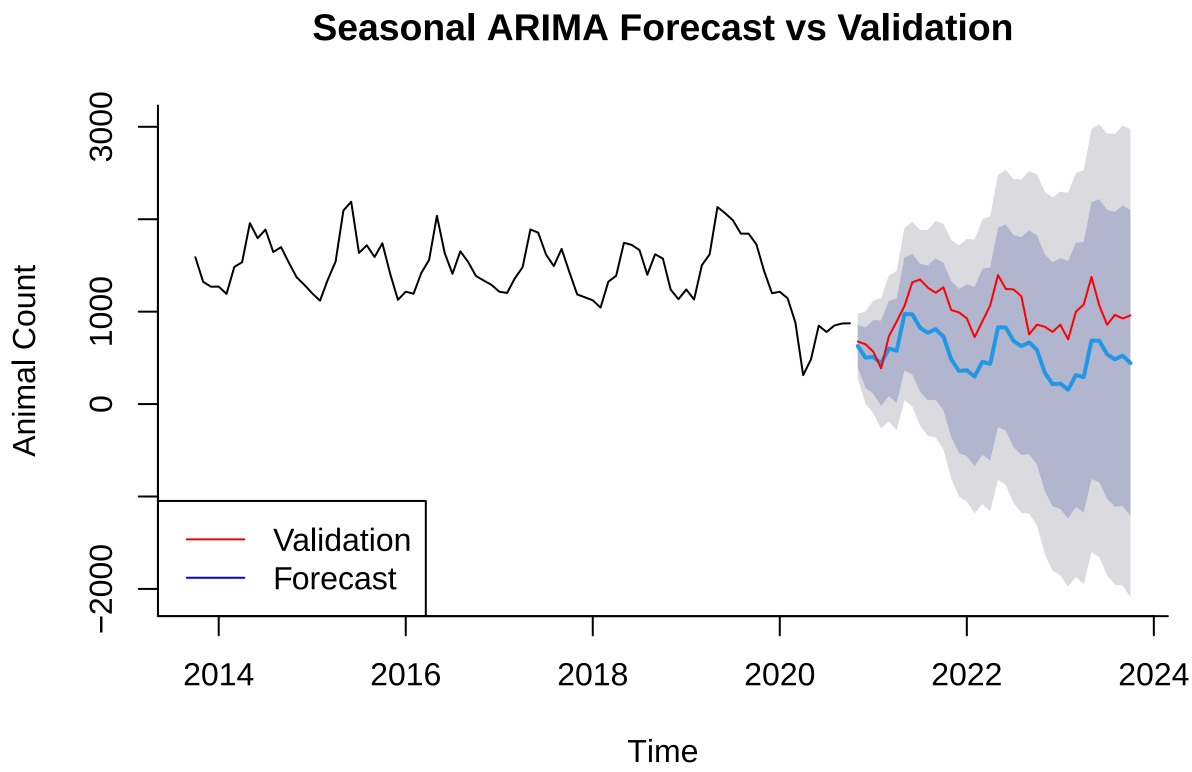
```r
train.ets.AAA.pred <- forecast(train.ets.AAA, h = nValid)
train.ets.MNM.pred <- forecast(train.ets.MNM, h = nValid)
train.ets.ANN.pred <- forecast(train.ets.ANN, h = nValid)
sarima_forecast <- forecast(sarima_model, h = nValid)

# Plot forecast against validation data
plot(train.ets.ANA.pred, main = "Model ETS(ANA) Forecast vs Validation", xlab = "Time", ylab = "Animal C
lines(valid.ts, col = "red")
legend("bottomleft", legend = c("Validation", "Forecast"), col = c("red", "blue"), lty = 1)
```



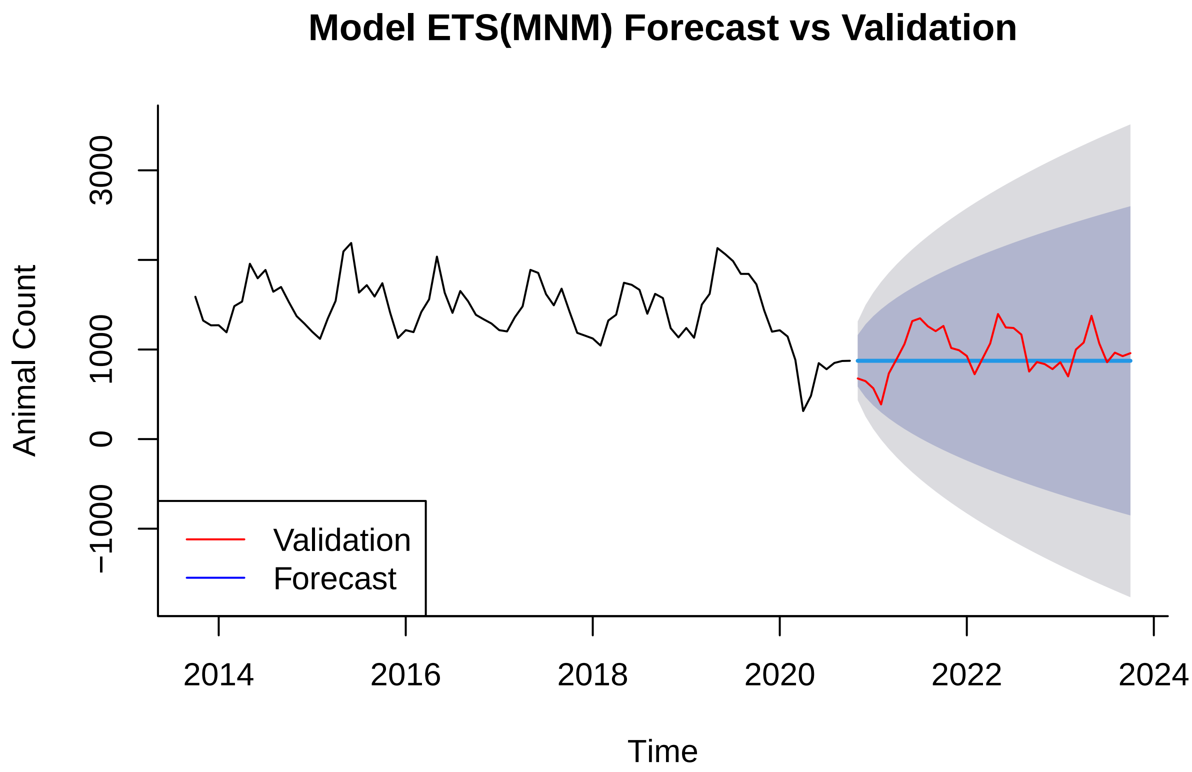

```r
plot(train.ets.AAA.pred, main = "Model ETS(AAA) Forecast vs Validation", xlab = "Time", ylab = "Animal C
lines(valid.ts, col = "red")
legend("bottomleft", legend = c("Validation", "Forecast"), col = c("red", "blue"), lty = 1)
```



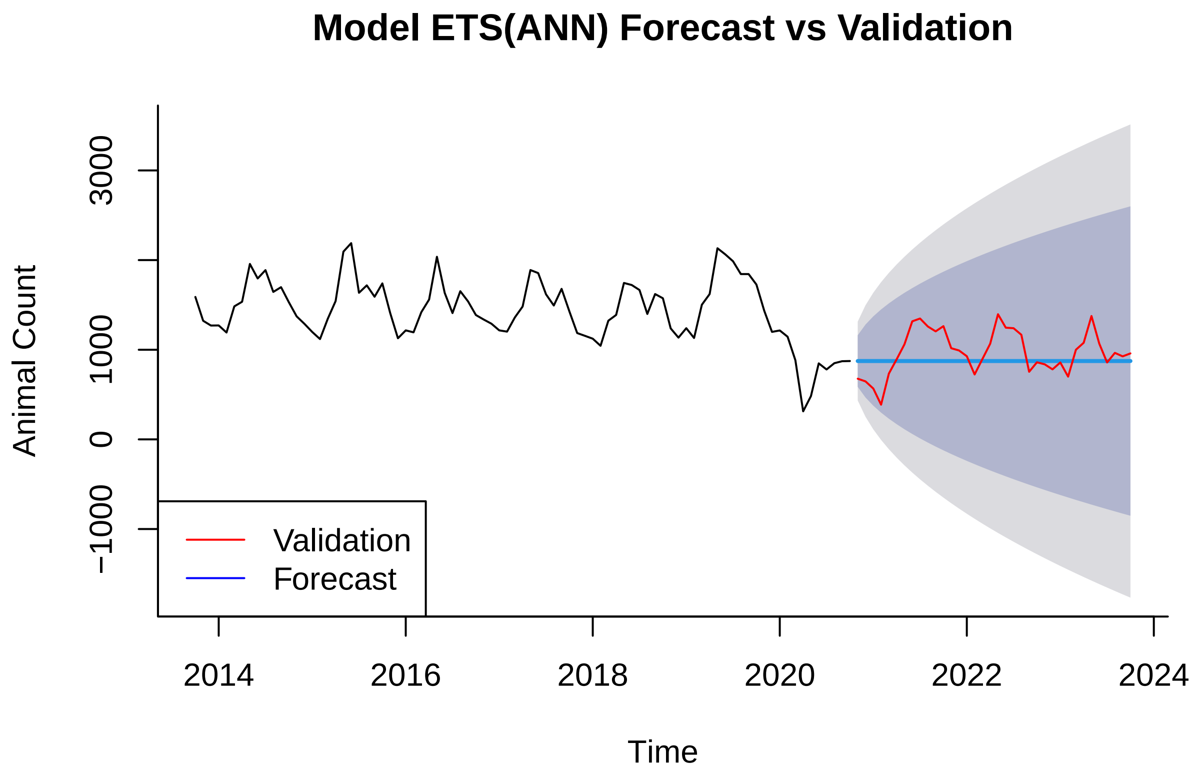

```
plot(sarima_forecast, main = "Seasonal ARIMA Forecast vs Validation", xlab = "Time", ylab = "Animal Cou
lines(valid.ts, col = "red")
legend("bottomleft", legend = c("Validation", "Forecast"), col = c("red", "blue"), lty = 1)
```




```r
plot(train.ets.MNM.pred, main = "Model ETS(MNM) Forecast vs Validation", xlab = "Time", ylab = "Animal C
lines(valid.ts, col = "red")
legend("bottomleft", legend = c("Validation", "Forecast"), col = c("red", "blue"), lty = 1)
```




```
plot(train.ets.ANN.pred, main = "Model ETS(ANN) Forecast vs Validation", xlab = "Time", ylab = "Animal C
lines(valid.ts, col = "red")
legend("bottomleft", legend = c("Validation", "Forecast"), col = c("red", "blue"), lty = 1)
```



Since our series does not have a consistent upward or downward trend due to the dip in 2020, the selected method was Exponential smoothing model to capture seasonality and proper forecast.

```
# Running accuracy metrics
cat("ETS(ANA)")
```

```
## ETS(ANA)
```

```
accuracy(train.ets.ANA.pred, valid.ts)
```

```
##                      ME     RMSE      MAE      MPE     MAPE      MASE
## Training set  -8.623941 158.1544 112.9657 -2.93292 10.59744 0.4698589
## Test set     134.772708 219.3923 185.3041 12.36179 19.14136 0.7707369
##                      ACF1 Theil's U
## Training set -0.001565088        NA
## Test set      0.527880678 0.9550673
```

```
cat("\n")
```

```
cat("ETS(AAA)")
```

```
## ETS(AAA)
```

```
accuracy(train.ets.AAA.pred, valid.ts)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  -8.582902 156.2808 108.2673 -3.021583 10.38140 0.4503168
## Test set     151.923886 229.1902 195.5521 14.129208 20.17142 0.8133612
##                    ACF1 Theil's U
## Training set 0.02587643        NA
```

```
## Test set     0.54420014 0.9974348
```

```r
cat("\n")
```

```r
cat("ETS(MNM)")
```

```
## ETS(MNM)
```

```r
accuracy(train.ets.MNM.pred, valid.ts)
```

```
##                     ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
## Training set -8.41333 221.7802 171.6125 -2.786877 13.70623 0.7137892 0.07020134
## Test set     99.66687 261.1778 209.5556  3.204536 22.44998 0.8716062 0.70483177
##              Theil's U
## Training set        NA
## Test set      1.195982
```

```r
cat("\n")
```

```r
cat("ETS(ANN)")
```

```
## ETS(ANN)
```

```r
accuracy(train.ets.ANN.pred, valid.ts)
```

```
##                     ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
## Training set -8.41333 221.7802 171.6125 -2.786877 13.70623 0.7137892 0.07020134
## Test set     99.66687 261.1778 209.5556  3.204536 22.44998 0.8716062 0.70483177
##              Theil's U
## Training set        NA
## Test set      1.195982
```

```r
cat("\n")
```

```r
cat("ETS(ARIMA)")
```

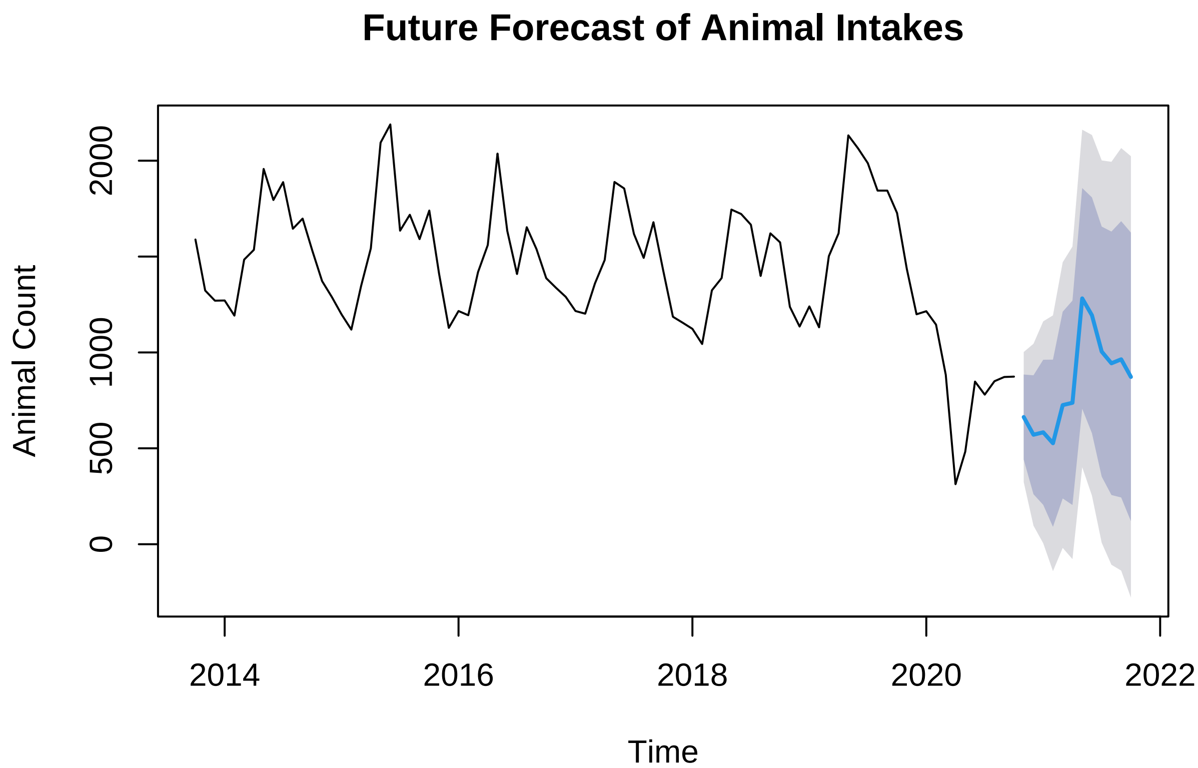
```
## ETS(ARIMA)
```

```r
accuracy(sarima_forecast, valid.ts)
```

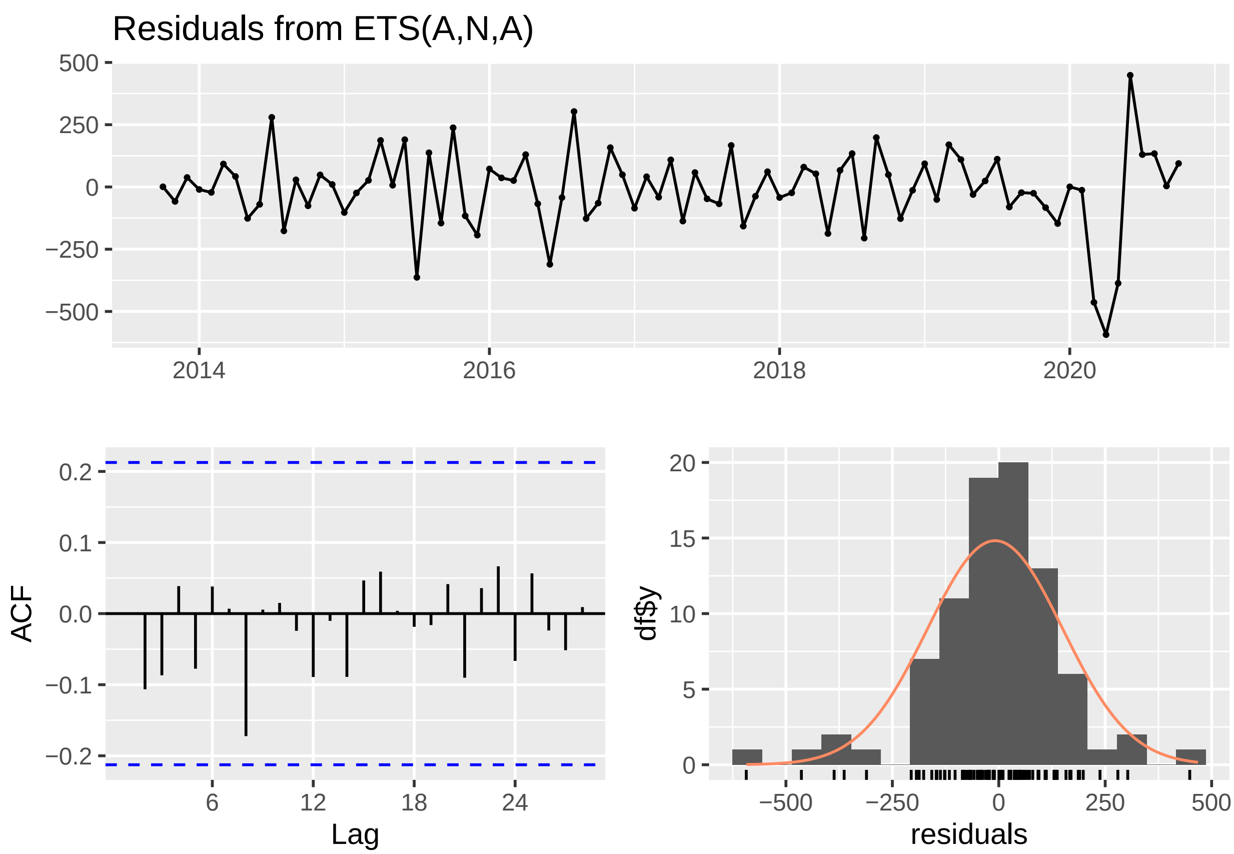
```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  -7.256432 162.6116 104.0028 -3.004169 10.14915 0.4325794
## Test set     421.775519 468.9000 424.8521 42.036736 42.83171 1.7670903
##                     ACF1 Theil's U
## Training set -0.01091148        NA
## Test set      0.65745235  2.124973
```

This results indicates *ETS(ANA)** is the best fit and forecast accuracy because it has the lowest RMSE, MAE, and MAPE.

```r
future_forecast <- forecast(train.ets.ANA, h = 12)
plot(future_forecast, main = "Future Forecast of Animal Intakes", xlab = "Time", ylab = "Animal Count")
```




```
checkresiduals(train.ets.ANA.pred)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,A)
## Q* = 7.7017, df = 17, p-value = 0.9725
##
## Model df: 0.   Total lags used: 17
```

In the time series of residuals, the residuals fluctuate around zero suggesting that the model has captured the main structure of the data.

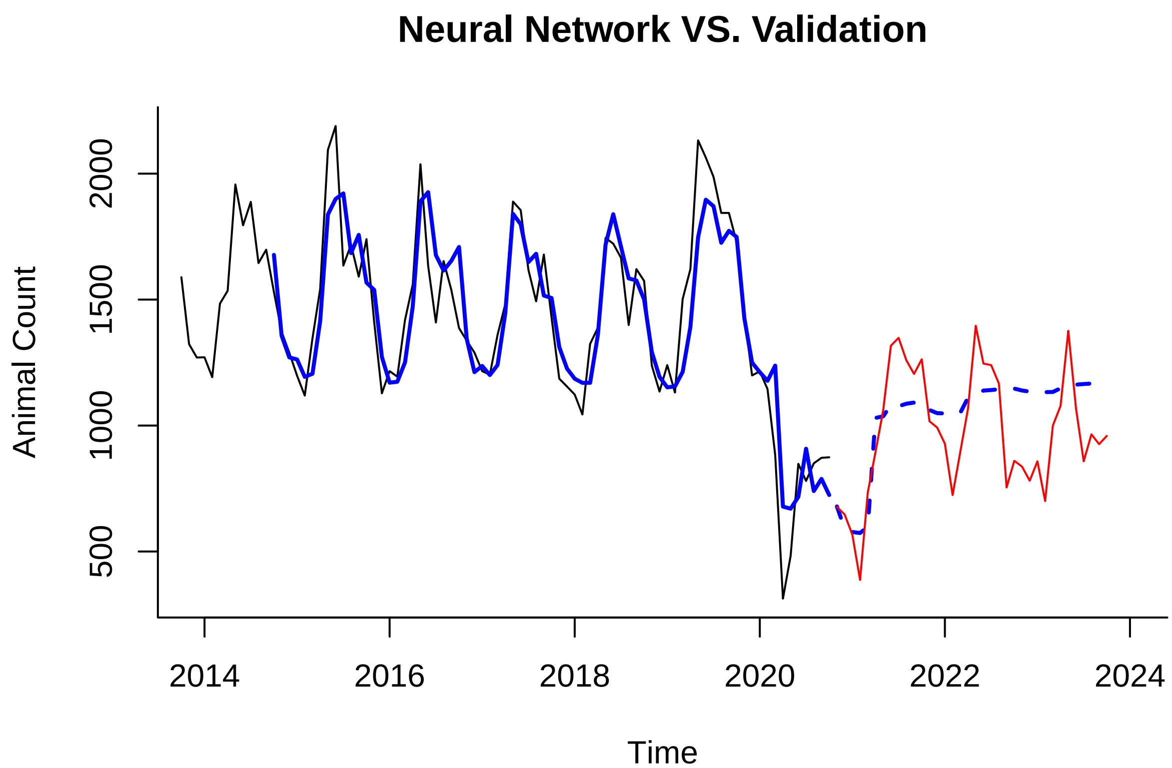
The ACF plot indicates the model has no significant autocorrelation because all the bars are withing the blue dashed lines.

The histogram shows a normal distribution.

The Ljung-Box test fails to reject the null hypothesis meaning the model explains the data.

```
# Neural Network Model
set.seed(201)

# Training model
train.nnetar <- nnetar(train.ts)

# Forecast the model
train.nnetar.pred <- forecast(train.nnetar, h = length(valid.ts))

# Plot
plot(train.ts, ylab = "Animal Count", xlab = "Time", bty = "l", lty = 1, main = "Neural Network VS. Vali
```

```r
lines(train.nnetar.pred$fitted, lwd = 2, col = "blue")

lines(train.nnetar.pred$mean, lwd = 2, col = "blue", lty = 2)

lines(valid.ts, col = "red")
```




```r
# Summary
cat("Summary:\n")
```

```
## Summary:
```

```r
summary(train.nnetar)
```

```
##           Length Class        Mode
## x         85     ts           numeric
## m          1     -none-       numeric
## p          1     -none-       numeric
## P          1     -none-       numeric
## scalex     2     -none-       list
## size       1     -none-       numeric
## subset    85     -none-       numeric
## model     20     nnetarmodels list
## nnetargs   0     -none-       list
## fitted    85     ts           numeric
## residuals 85     ts           numeric
## lags       2     -none-       numeric
## series     1     -none-       character
```

```
## method     1      -none-       character
## call       2      -none-       call
```

```r
# Checking accuracy
cat("\nAccuracy:")
```

```
##
## Accuracy:
```

```r
accuracy(train.nnetar.pred, valid.ts)
```

```
##                         ME     RMSE      MAE        MPE     MAPE      MASE
## Training set   0.05186156 151.8806 118.9752  -2.259519 10.07721 0.4948543
## Test set     -68.82465162 207.7488 174.3936 -10.842267 19.62712 0.7253567
##                   ACF1 Theil's U
## Training set 0.2845750        NA
## Test set     0.5860013  1.004827
```