Foodborne Illness Time Series Forecasting

This project forecasts monthly foodborne illness counts using time series models like ARIMA, Prophet and machine learning Random Forest. Evaluation include the RMSE, MAE, change-point detection, lead time analysis, and outbreak classification accuracy.

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
food_data_clean <- read_csv("/Users/mruduladeshmukh/Downloads/outbreaks_clean (1).csv",
show_col_types = FALSE)

glimpse(food_data_clean)
```

```
## Rows: 12,520
## Columns: 14
## $ Year                <dbl> 1998, 1998, 1998, 1998, 1998, 1998, 1998, 1998, 19…
## $ Month               <chr> "April", "April", "April", "April", "April", "Apri…
## $ State               <chr> "California", "California", "California", "Califor…
## $ Location            <chr> "Catering Service", "Restaurant", "Restaurant", "R…
## $ Illnesses           <dbl> 11, 7, 11, 19, 72, 3, 12, 2, 3, 3, 3, 4, 5, 6, 5, …
## $ Hospitalizations    <dbl> 0, 0, 0, 0, 7, 0, 2, 0, 0, 1, 2, 0, 0, 0, 0, 0, 1,…
## $ Fatalities          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
## $ Month_Num           <dbl> 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,…
## $ Date                <date> 1998-04-01, 1998-04-01, 1998-04-01, 1998-04-01, 1…
## $ Food                <chr> NA, "Pizza, Meat", "Pate, Unspecified", NA, NA, "S…
## $ Ingredient          <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA…
## $ Species             <chr> NA, NA, "Clostridium perfringens", NA, "Shigella b…
## $ `Serotype/Genotype` <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, "Enter…
## $ Status              <chr> NA, NA, "Confirmed", NA, "Confirmed", "Suspected",…
```

```
sum(food_data_clean$Illnesses == 0)
```

```
## [1] 0
```

```
food_data_clean <- food_data_clean %>%
  filter(Illnesses > 0)
```

```
monthly_illness <- food_data_clean %>%
  group_by(Date) %>%
  summarise(Total_Illnesses = sum(Illnesses, na.rm = TRUE)) %>%
  arrange(Date)
```

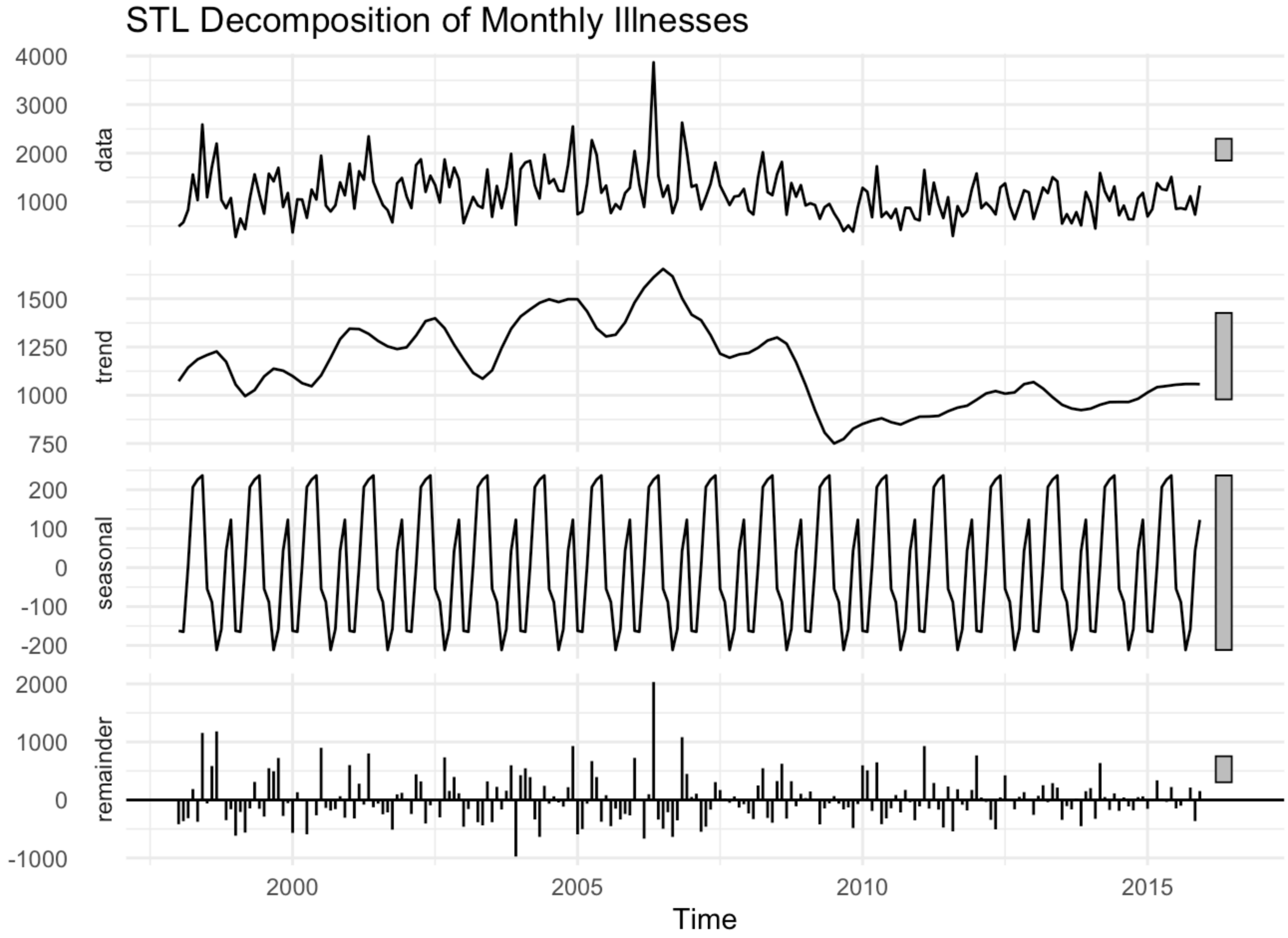
```
illness_ts <- ts(monthly_illness$Total_Illnesses,
                 start = c(year(min(monthly_illness$Date)), month(min(monthly_illness$Da
te))),
                 frequency = 12)
```

```
head(monthly_illness)
```

```
## # A tibble: 6 × 2
##   Date       Total_Illnesses
##   <date>               <dbl>
## 1 1998-01-01             492
## 2 1998-02-01             578
## 3 1998-03-01             833
## 4 1998-04-01            1560
## 5 1998-05-01            1035
## 6 1998-06-01            2589
```

```
decomp <- stl(illness_ts, s.window = "periodic")
```

```
autoplot(decomp) +
  ggtitle("STL Decomposition of Monthly Illnesses") +
  theme_minimal()
```



```
adf_test <- adf.test(illness_ts)
```

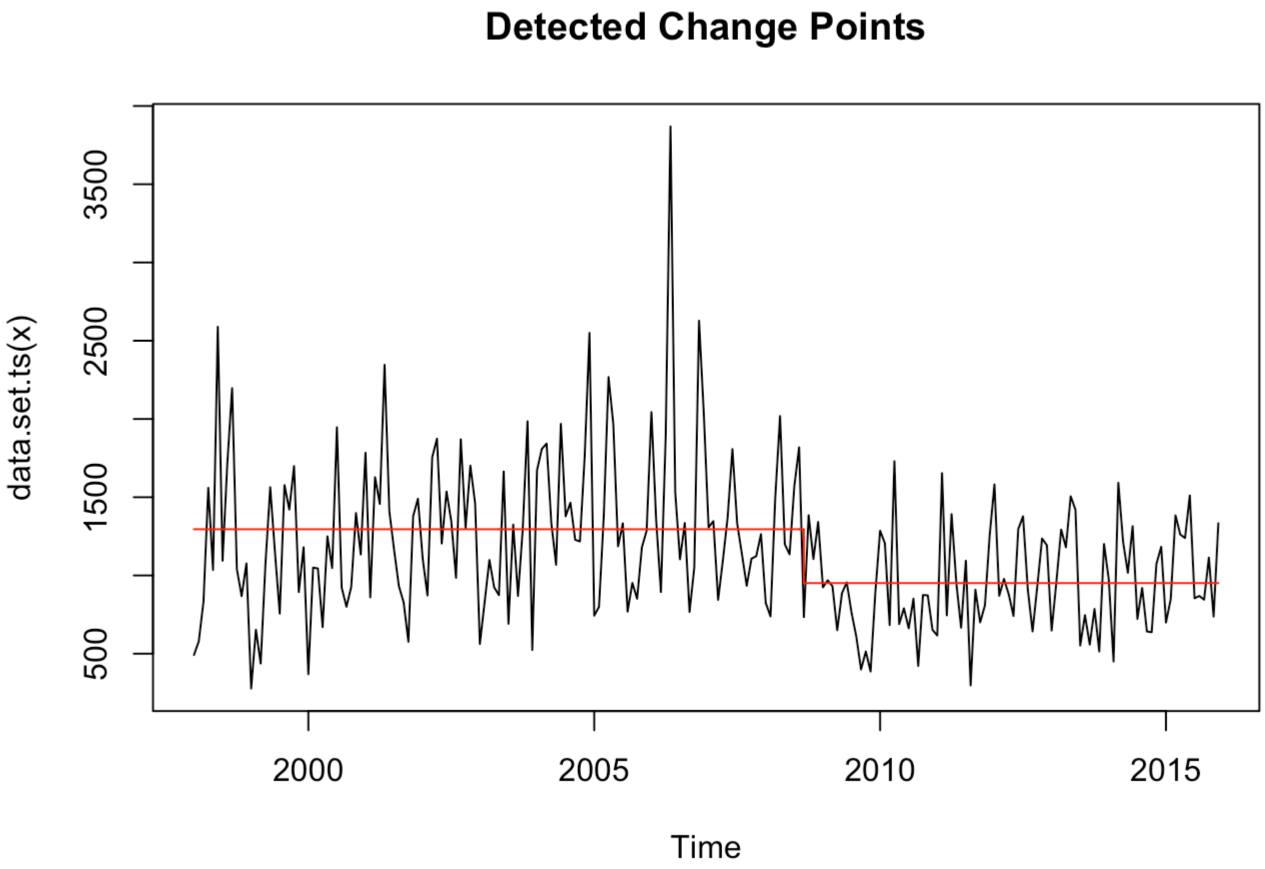
```
## Warning in adf.test(illness_ts): p-value smaller than printed p-value
```

```
print(adf_test)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  illness_ts
## Dickey-Fuller = -4.7126, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

We apply the PELT algorithm using the `changepoint` package to detect structural changes (e.g., outbreaks or shifts) in the time series. For each segment between change points, we fit a separate ARIMA model to understand the local behavior.

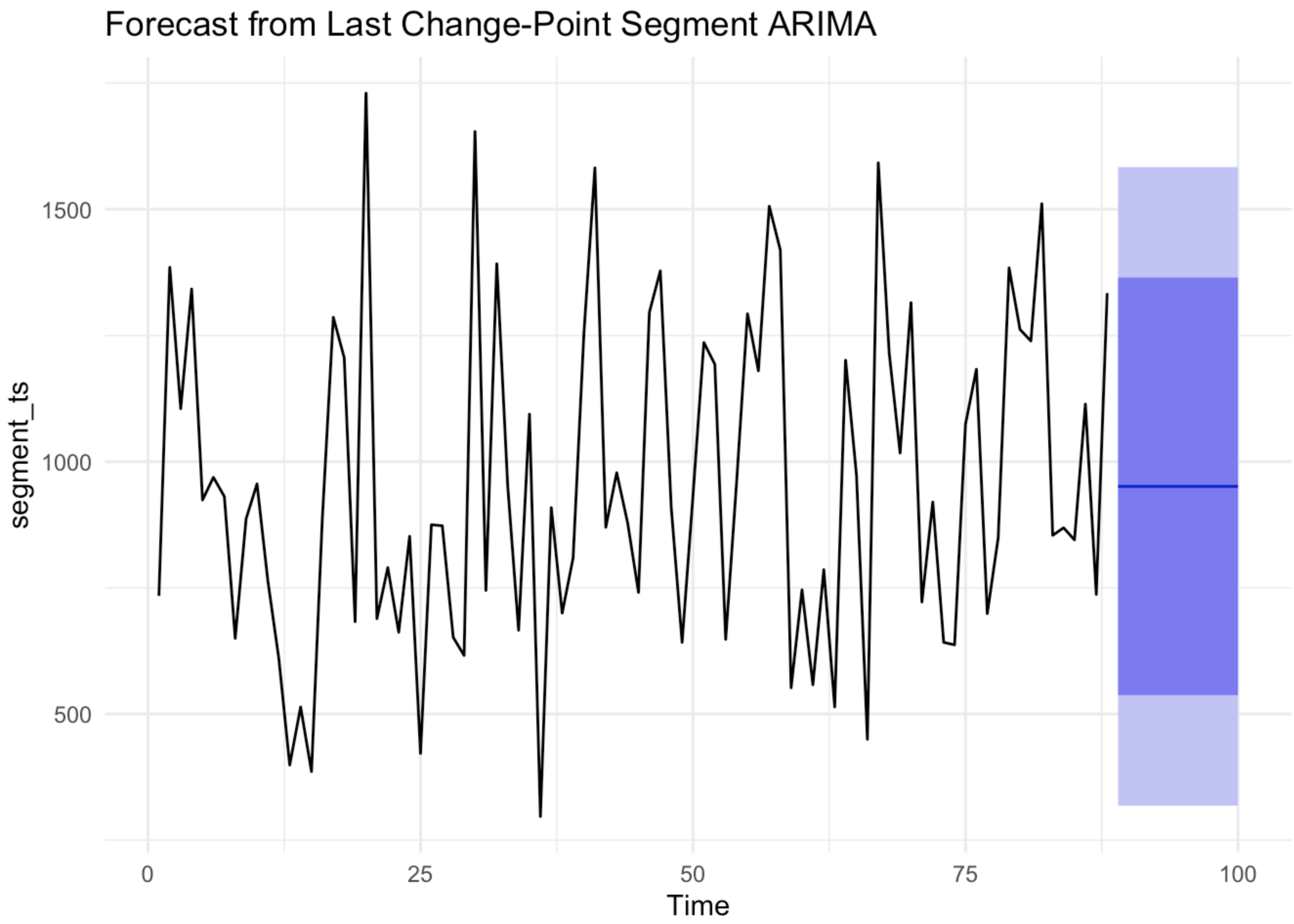
```
cpt_result <- cpt.meanvar(illness_ts, method = "PELT")
plot(cpt_result, main = "Detected Change Points")
```



```
cp_indices <- c(0, cpts(cpt_result), length(illness_ts))
segment_models <- list()
for (i in 1:(length(cp_indices) - 1)) {
  start_idx <- cp_indices[i] + 1
  end_idx <- cp_indices[i + 1]

  if (start_idx >= end_idx) {
    cat(paste0("⚠️ Skipping segment ", i, ": invalid index (", start_idx, " > ", end_idx, ")\n"))
    next
  }

  segment_ts <- illness_ts[start_idx:end_idx]

  model <- auto.arima(segment_ts)
  segment_models[[i]] <- model

  cat(paste0("Segment ", i, ": Index ", start_idx, " to ", end_idx, "\n"))
  print(summary(model))
}
```

```
## Segment 1: Index 1 to 128
## Series: segment_ts 
## ARIMA(0,0,1) with non-zero mean 
## 
## Coefficients:
##          ma1       mean
##       0.2112  1294.9513
## s.e.  0.0874    54.9071
## 
## sigma^2 = 267952:  log likelihood = -980.55
## AIC=1967.09   AICc=1967.29   BIC=1975.65
## 
## Training set error measures:
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.9916662 513.5808 390.7947 -17.92213 37.48717 0.7545103
##                     ACF1
## Training set -0.00508288
## Segment 2: Index 129 to 216
## Series: segment_ts 
## ARIMA(0,0,0) with non-zero mean 
## 
## Coefficients:
##           mean
##       951.0682
## s.e.   34.2046
## 
## sigma^2 = 104139:  log likelihood = -632.72
## AIC=1269.43   AICc=1269.58   BIC=1274.39
## 
## Training set error measures:
##                         ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 3.888607e-13 320.8671 264.1457 -13.83765 33.16844 0.7767414
##                    ACF1
## Training set 0.09906248
```

```
final_model <- segment_models[[length(segment_models)]]
forecast_last <- forecast(final_model, h = 12)

autoplot(forecast_last) +
  ggtitle("Forecast from Last Change-Point Segment ARIMA") +
  theme_minimal()
```

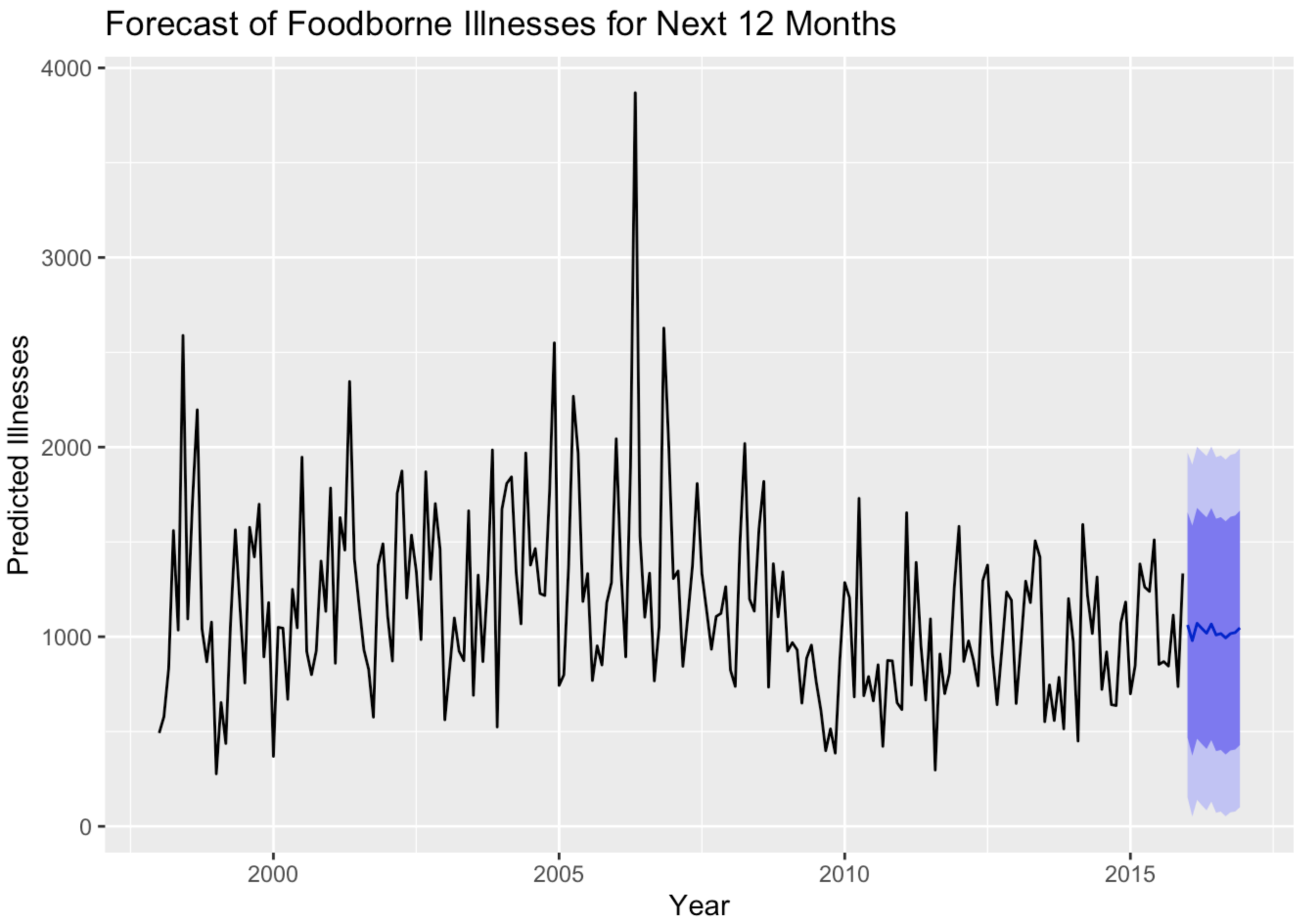


```
arima_model <- auto.arima(illness_ts)
summary(arima_model)
```

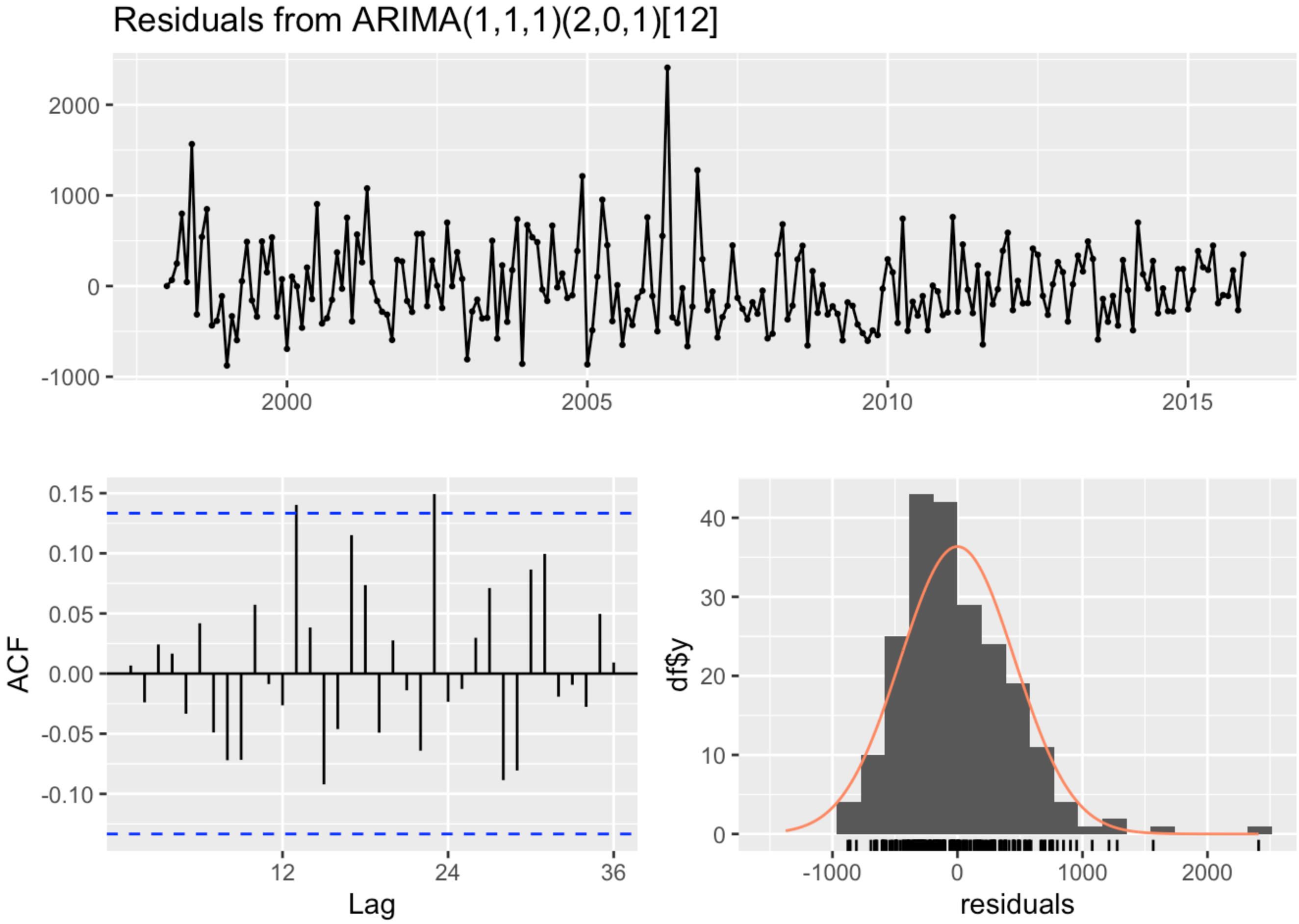
```
## Series: illness_ts
## ARIMA(1,1,1)(2,0,1)[12]
##
## Coefficients:
##          ar1      ma1      sar1    sar2    sma1
##       0.1540  -0.9489  -0.4731  0.0922  0.5486
## s.e.  0.0734   0.0239   0.4877  0.0766  0.4846
##
## sigma^2 = 214871:  log likelihood = -1623.45
## AIC=3258.89   AICc=3259.3   BIC=3279.12
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.1703716 457.0585 352.8889 -15.75537 36.15144 0.7643398
##                    ACF1
## Training set 0.00671663
```

```
forecast_illness <- forecast(arima_model, h = 12)
```

```
autoplot(forecast_illness) +
  ggtitle("Forecast of Foodborne Illnesses for Next 12 Months") +
  xlab("Year") +
  ylab("Predicted Illnesses")
```



```
checkresiduals(arima_model)
```



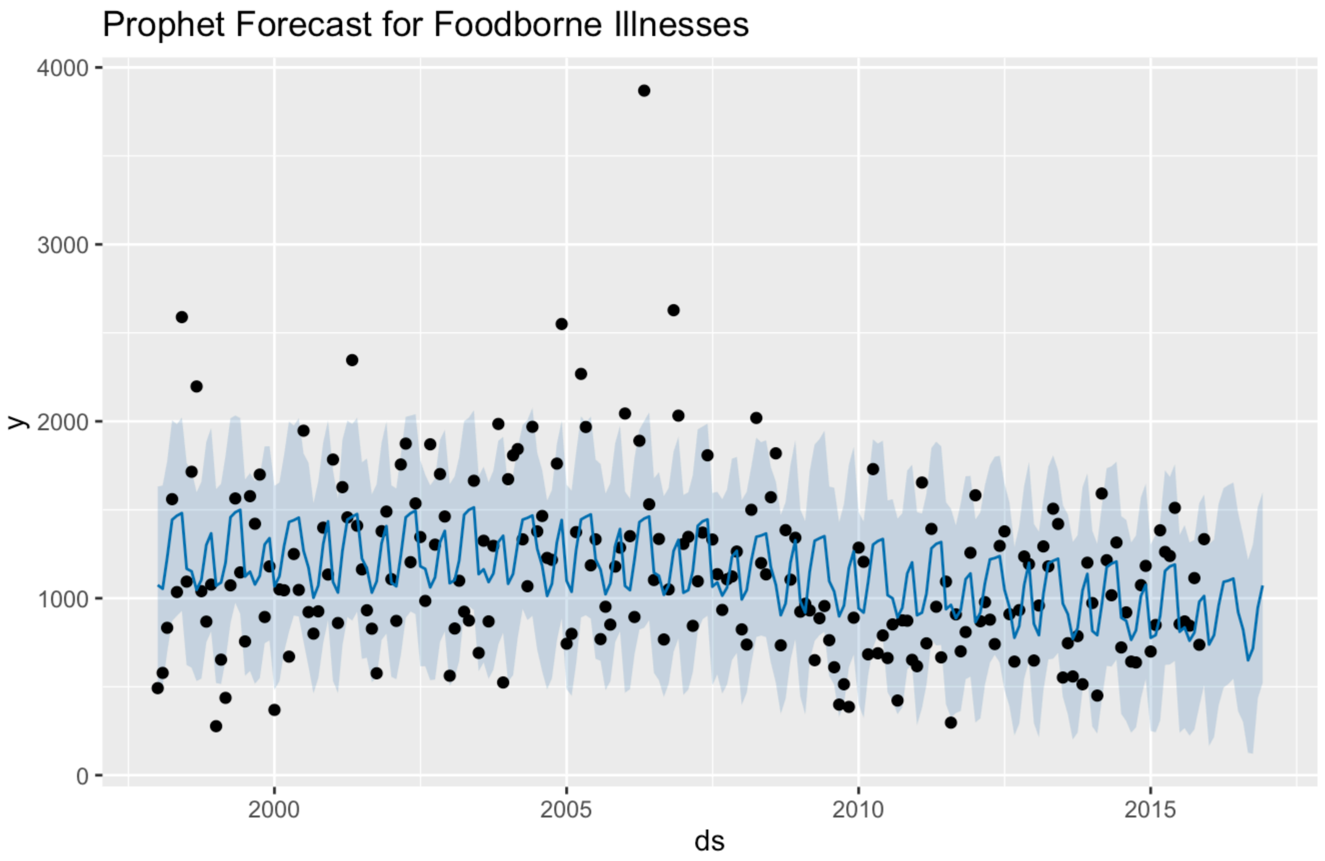
```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(2,0,1)[12]
## Q* = 23.983, df = 19, p-value = 0.1968
##
## Model df: 5.   Total lags used: 24
```

```
prophet_df <- monthly_illness %>%
  rename(ds = Date, y = Total_Illnesses)
```

```
m <- prophet(prophet_df, weekly.seasonality = FALSE, daily.seasonality = FALSE)
```

```
future <- make_future_dataframe(m, periods = 12, freq = "month")
forecast_prophet <- predict(m, future)
```

```
plot(m, forecast_prophet) +
  ggtitle("Prophet Forecast for Foodborne Illnesses")
```



```
actual_df <- monthly_illness %>%
  rename(ds = Date, y = Total_Illnesses) %>%
  mutate(model = "Actual")
```

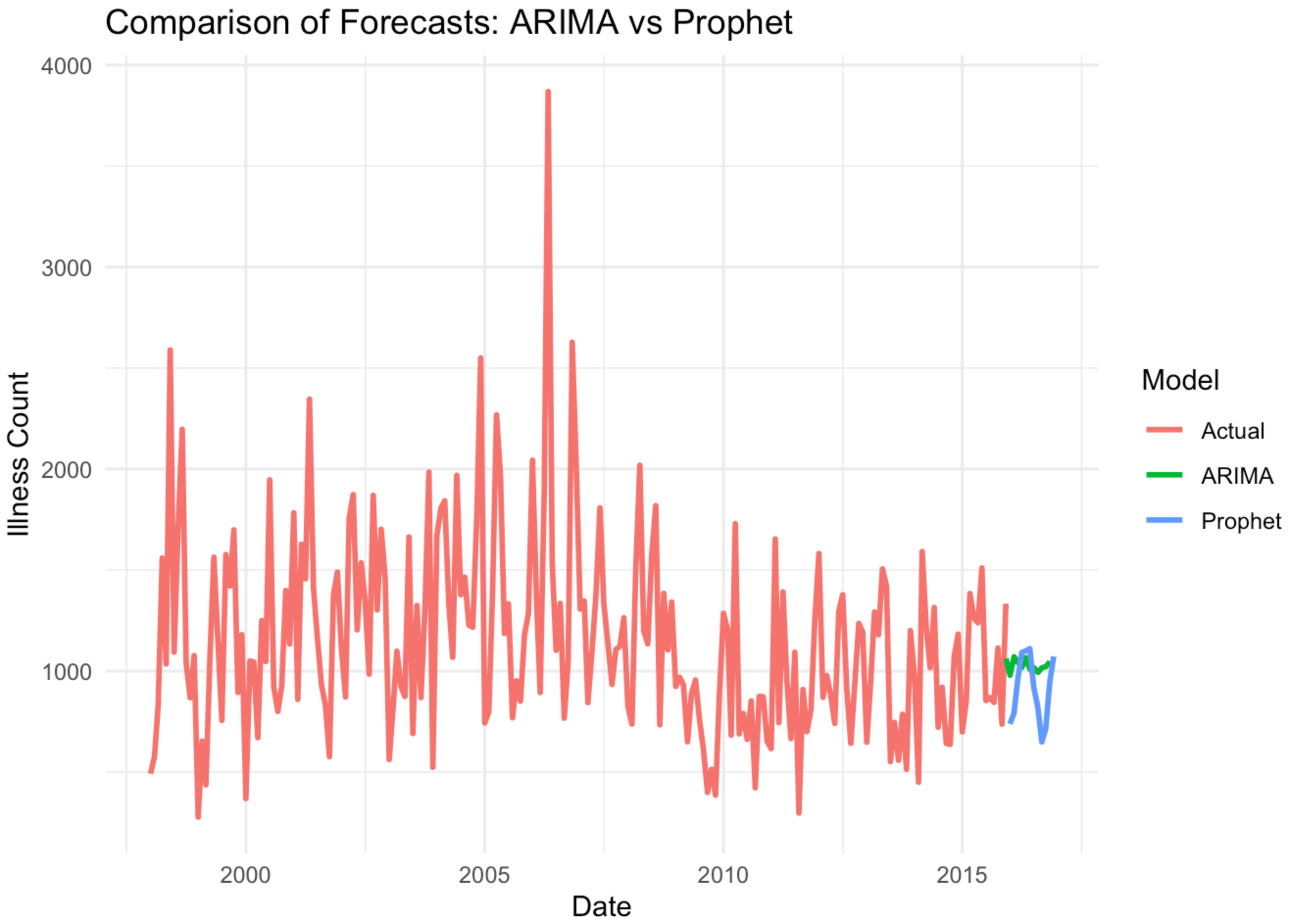
```
arima_forecast <- forecast(arima_model, h = 12)
arima_df <- data.frame(
  ds = seq(max(monthly_illness$Date) + 1, by = "month", length.out = 12),
  y = as.numeric(arima_forecast$mean),
  model = "ARIMA"
)
```

```
prophet_df_future <- forecast_prophet %>%
  filter(as.Date(ds) > max(as.Date(actual_df$ds))) %>%
  select(ds, y = yhat) %>%
  mutate(model = "Prophet")
```

```
combined <- bind_rows(actual_df, arima_df, prophet_df_future)
```

```
ggplot(combined, aes(x = ds, y = y, color = model)) +
  geom_line(size = 1) +
  labs(title = "Comparison of Forecasts: ARIMA vs Prophet",
       x = "Date", y = "Illness Count", color = "Model") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
train_data <- head(monthly_illness, -12)
test_data <- tail(monthly_illness, 12)
test_values <- test_data$Total_Illnesses
```

```
illness_ts_train <- ts(train_data$Total_Illnesses,
                       start = c(year(min(train_data$Date)), month(min(train_data$Dat
e))),
                       frequency = 12)
```

```
arima_model <- auto.arima(illness_ts_train)
arima_forecast <- forecast(arima_model, h = 12)
arima_pred <- as.numeric(arima_forecast$mean)
```

```
prophet_train <- train_data %>%
  rename(ds = Date, y = Total_Illnesses)
```

```
m <- prophet(prophet_train, weekly.seasonality = FALSE, daily.seasonality = FALSE)
```

```
future <- make_future_dataframe(m, periods = 12, freq = "month")
forecast_prophet <- predict(m, future)
```

```
prophet_pred <- tail(forecast_prophet$yhat, 12)
```

```
metrics_df <- tibble(
  truth = test_values,
  .pred = arima_pred
)
```

```
rmse_arima <- rmse(metrics_df, truth = truth, estimate = .pred)
mae_arima  <- mae(metrics_df, truth = truth, estimate = .pred)
```

```
metrics_prophet <- tibble(
  truth = test_values,
  .pred = prophet_pred
)
rmse_prophet <- rmse(metrics_prophet, truth = truth, estimate = .pred)
mae_prophet <- mae(metrics_prophet, truth = truth, estimate = .pred)
```

```
print(rmse_arima)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        269.
```

```
print(mae_arima)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 mae     standard        237.
```

```
print(rmse_prophet)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        277.
```

```
print(mae_prophet)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 mae     standard        226.
```

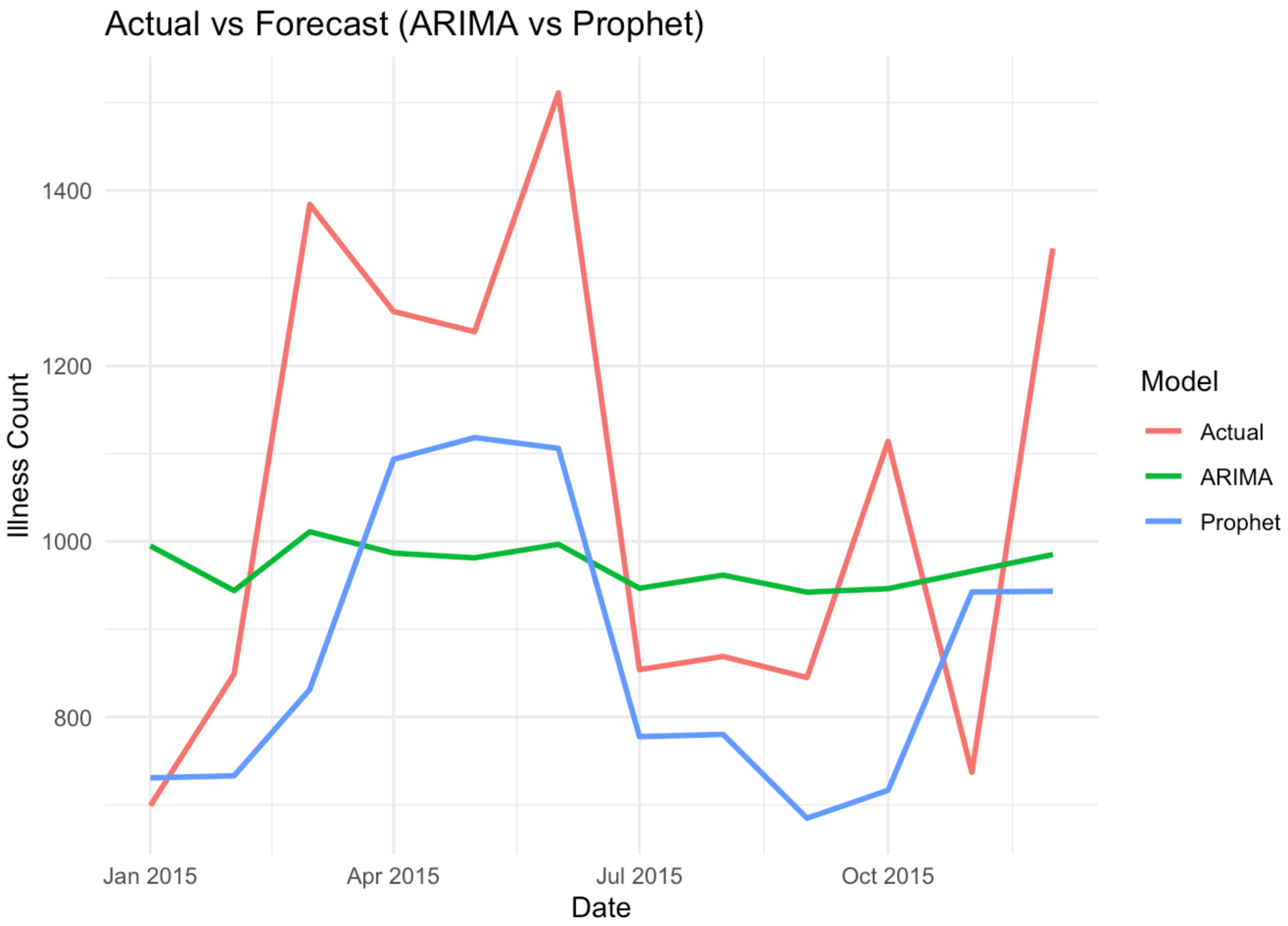
```
arima_peak_index <- which.max(arima_pred)
prophet_peak_index <- which.max(prophet_pred)

actual_peak_index <- which.max(test_values)

lead_time_arima <- actual_peak_index - arima_peak_index
lead_time_prophet <- actual_peak_index - prophet_peak_index

lead_time_tbl <- tibble(
  Model = c("ARIMA", "Prophet"),
  Predicted_Peak = c(arima_peak_index, prophet_peak_index),
  Actual_Peak = actual_peak_index,
  Lead_Time = c(lead_time_arima, lead_time_prophet)
)
print(lead_time_tbl)
```

```
## # A tibble: 2 × 4
##   Model   Predicted_Peak Actual_Peak Lead_Time
##   <chr>            <int>       <int>     <int>
## 1 ARIMA                3           6         3
## 2 Prophet              5           6         1
```

```
compare_df <- data.frame(
  Date = test_data$Date,
  Actual = test_values,
  ARIMA = arima_pred,
  Prophet = prophet_pred
)
```

```
compare_df_long <- compare_df %>%
  pivot_longer(cols = c("Actual", "ARIMA", "Prophet"), names_to = "Model", values_to =
"Value")
```

```
ggplot(compare_df_long, aes(x = Date, y = Value, color = Model)) +
  geom_line(size = 1) +
  labs(title = "Actual vs Forecast (ARIMA vs Prophet)", y = "Illness Count") +
  theme_minimal()
```



```
threshold <- mean(test_values) + sd(test_values)
classification_df <- tibble(
  truth = ifelse(test_values > threshold, 1, 0),
  ARIMA = ifelse(arima_pred > threshold, 1, 0),
  Prophet = ifelse(prophet_pred > threshold, 1, 0)
)

classification_long <- classification_df %>%
  pivot_longer(cols = c(ARIMA, Prophet), names_to = "Model", values_to = ".pred_class")
%>%
  mutate(
    .truth = factor(truth, levels = c(0, 1)),
    .pred_class = factor(.pred_class, levels = c(0, 1))
  )

classification_metrics <- classification_long %>%
  group_by(Model) %>%
  summarise(
    Sensitivity = sensitivity_vec(.truth, .pred_class),
    Specificity = specificity_vec(.truth, .pred_class),
    Accuracy = accuracy_vec(.truth, .pred_class)
  )
print(classification_metrics)
```

```
## # A tibble: 2 × 4
##   Model   Sensitivity Specificity Accuracy
##   <chr>         <dbl>       <dbl>    <dbl>
## 1 ARIMA             1           0    0.833
## 2 Prophet           1           0    0.833
```

```
df_prophet <- monthly_illness %>%

  select(Date, Total_Illnesses) %>%
  rename(ds = Date, y = Total_Illnesses)
```

```
m <- prophet(df_prophet, weekly.seasonality = FALSE, daily.seasonality = FALSE)
future <- make_future_dataframe(m, periods = 12, freq = "month")
forecast_prophet <- predict(m, future)
```

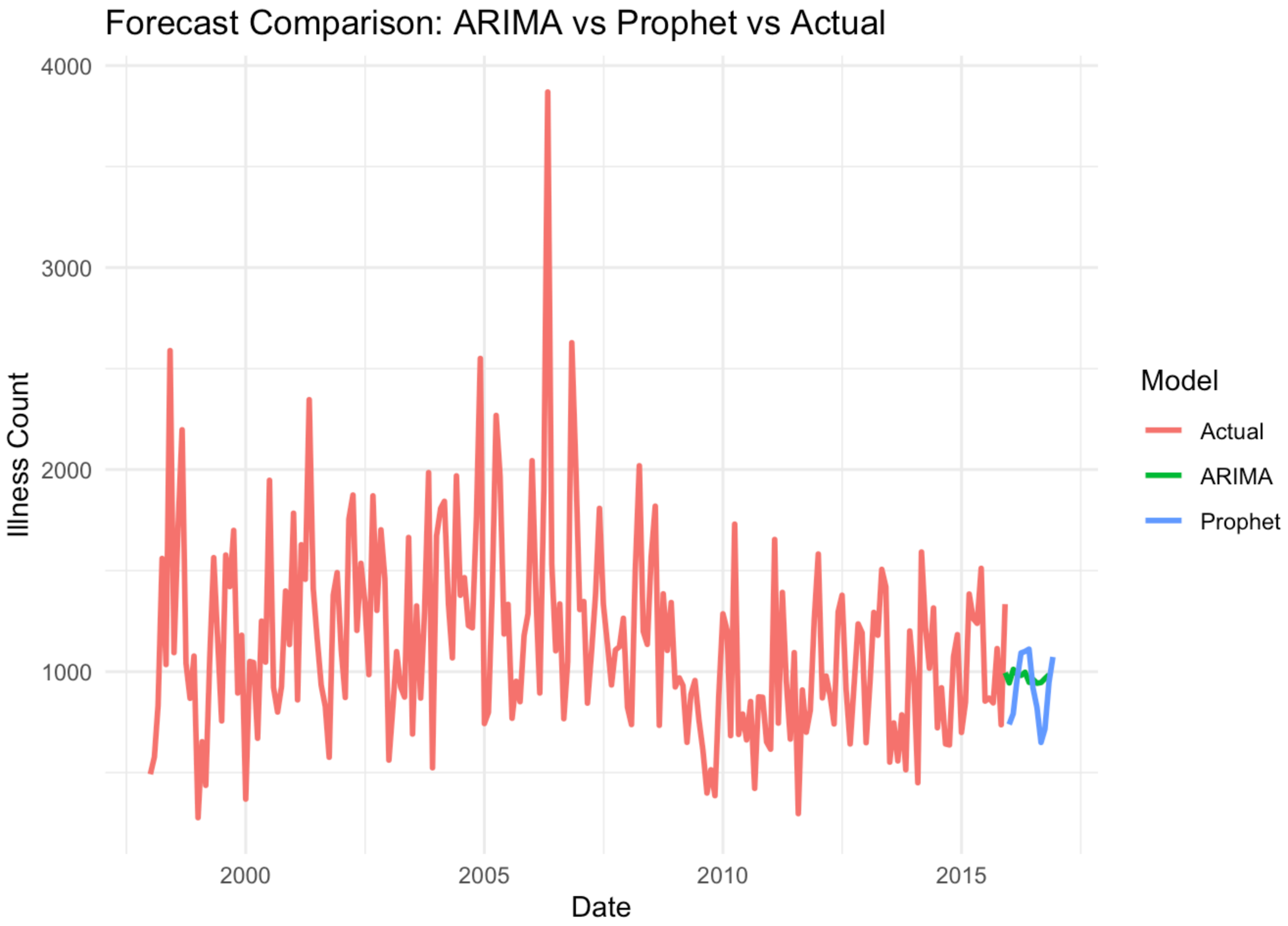
```
prophet_df_forecast <- forecast_prophet %>%
  tail(12) %>%
  select(ds, yhat) %>%
  mutate(ds = floor_date(as.Date(ds), unit = "month"), model = "Prophet") %>%
  rename(y = yhat)
```

```
actual_df <- df_prophet %>%
  mutate(model = "Actual")
```

```
forecast_arima <- forecast(arima_model, h = 12)
arima_forecast_df <- data.frame(
  ds = seq(from = as.Date(max(df_prophet$ds)) + 1, by = "month", length.out = 12),
  y = as.numeric(forecast_arima$mean),
  model = "ARIMA"
)
```

```
combined_df <- bind_rows(actual_df, arima_forecast_df, prophet_df_forecast)
```

```
ggplot(combined_df, aes(x = ds, y = y, color = model)) +
  geom_line(size = 1) +  # unified line styling for all models
  labs(title = "Forecast Comparison: ARIMA vs Prophet vs Actual",
       x = "Date", y = "Illness Count", color = "Model") +
  theme_minimal()
```



```
arima_df <- data.frame(
  ds = seq(from = as.Date("2015-01-01"), by = "month", length.out = 12),
  arima = as.numeric(arima_forecast$mean[1:12])
)
```

```
prophet_df_forecast <- forecast_prophet %>%
  mutate(ds = floor_date(as.Date(ds), unit = "month")) %>%
  filter(ds >= as.Date("2015-01-01") & ds <= as.Date("2015-12-01")) %>%
  group_by(ds) %>%
  summarise(prophet = mean(yhat)) %>%
  ungroup()
```

```
actual_df <- monthly_illness %>%
  mutate(ds = floor_date(Date, unit = "month")) %>%
  group_by(ds) %>%
  summarise(actual = sum(Total_Illnesses, na.rm = TRUE)) %>%
  ungroup() %>%
  filter(ds >= as.Date("2015-01-01") & ds <= as.Date("2015-12-01"))
```

```
compare_df <- actual_df %>%
  inner_join(arima_df, by = "ds") %>%
  inner_join(prophet_df_forecast, by = "ds")
```

```
print(compare_df)
```

```
## # A tibble: 12 × 4
##    ds         actual arima prophet
##    <date>      <dbl> <dbl>   <dbl>
##  1 2015-01-01    699  995.    776.
##  2 2015-02-01    849  944.    791.
##  3 2015-03-01   1384 1011.    900.
##  4 2015-04-01   1262  987.   1155.
##  5 2015-05-01   1239  981.   1180.
##  6 2015-06-01   1511  997.   1191.
##  7 2015-07-01    854  947.    811.
##  8 2015-08-01    869  962.    835.
##  9 2015-09-01    845  942.    760.
## 10 2015-10-01   1114  946.    805.
## 11 2015-11-01    737  966.    980.
## 12 2015-12-01   1334  985.   1013.
```

```
rmse_arima <- sqrt(mean((compare_df$arima - compare_df$actual)^2))
mae_arima  <- mean(abs(compare_df$arima - compare_df$actual))
```

```
rmse_prophet <- sqrt(mean((compare_df$prophet - compare_df$actual)^2))
mae_prophet  <- mean(abs(compare_df$prophet - compare_df$actual))
```

```
cat("ARIMA:   RMSE =", round(rmse_arima, 2), "  MAE =", round(mae_arima, 2), "\n")
```

```
## ARIMA:   RMSE = 269.45   MAE = 236.62
```

```
cat("Prophet: RMSE =", round(rmse_prophet, 2), "  MAE =", round(mae_prophet, 2), "\n")
```

```
## Prophet: RMSE = 229.04   MAE = 178.47
```

```
ml_df <- food_data_clean %>%
  select(Illnesses, Year, Month, State, Location, Species, Status) %>%
  filter(!is.na(Illnesses)) %>%
  mutate(across(where(is.character), as.factor))

ml_df <- ml_df %>%
  filter(Illnesses < quantile(Illnesses, 0.99))

skim(ml_df)
```

Data summary

Name	ml_df
Number of rows	12392
Number of columns	7
_______________________	
Column type frequency:	
factor	5
numeric	2
________________________	
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Month	0	1.00	FALSE	12	Dec: 1222, May: 1213, Jun: 1188, Apr: 1105
State	0	1.00	FALSE	55	Cal: 1609, Ill: 893, Ohi: 880, Flo: 840
Location	0	1.00	FALSE	141	Res: 7111, Pri: 1377, Cat: 860, Fas: 356
Species	3540	0.71	FALSE	262	Nor: 1900, Sal: 1617, Nor: 1180, Nor: 630
Status	3540	0.71	FALSE	25	Con: 5708, Sus: 2678, Sus: 199, Con: 94

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Illnesses	0	1	16.68	22.58	2	4	9	20	187	▇▁▁▁▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Year	0	1	2006.60	5.13	1998	2002	2006	2011	2015	▇▇▇▅▇

```
set.seed(123)
split <- initial_split(ml_df, prop = 0.8, strata = Illnesses)
train_data <- training(split)
test_data <- testing(split)
```

```
ml_recipe <- recipe(Illnesses ~ ., data = train_data) %>%
  step_impute_median(all_numeric_predictors()) %>%
  step_impute_mode(all_nominal_predictors()) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_zv(all_predictors())
```

```
rf_model <- rand_forest(mtry = 5, trees = 500, min_n = 10) %>%
  set_engine("randomForest") %>%
  set_mode("regression")
```

```
rf_workflow <- workflow() %>%
  add_recipe(ml_recipe) %>%
  add_model(rf_model)

rf_fit <- rf_workflow %>% fit(data = train_data)
```

```
rf_preds <- rf_fit %>%
  predict(new_data = test_data) %>%
  bind_cols(test_data)
metrics(rf_preds, truth = Illnesses, estimate = .pred)
```

```
## # A tibble: 3 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard      20.6
## 2 rsq     standard       0.209
## 3 mae     standard      12.8
```

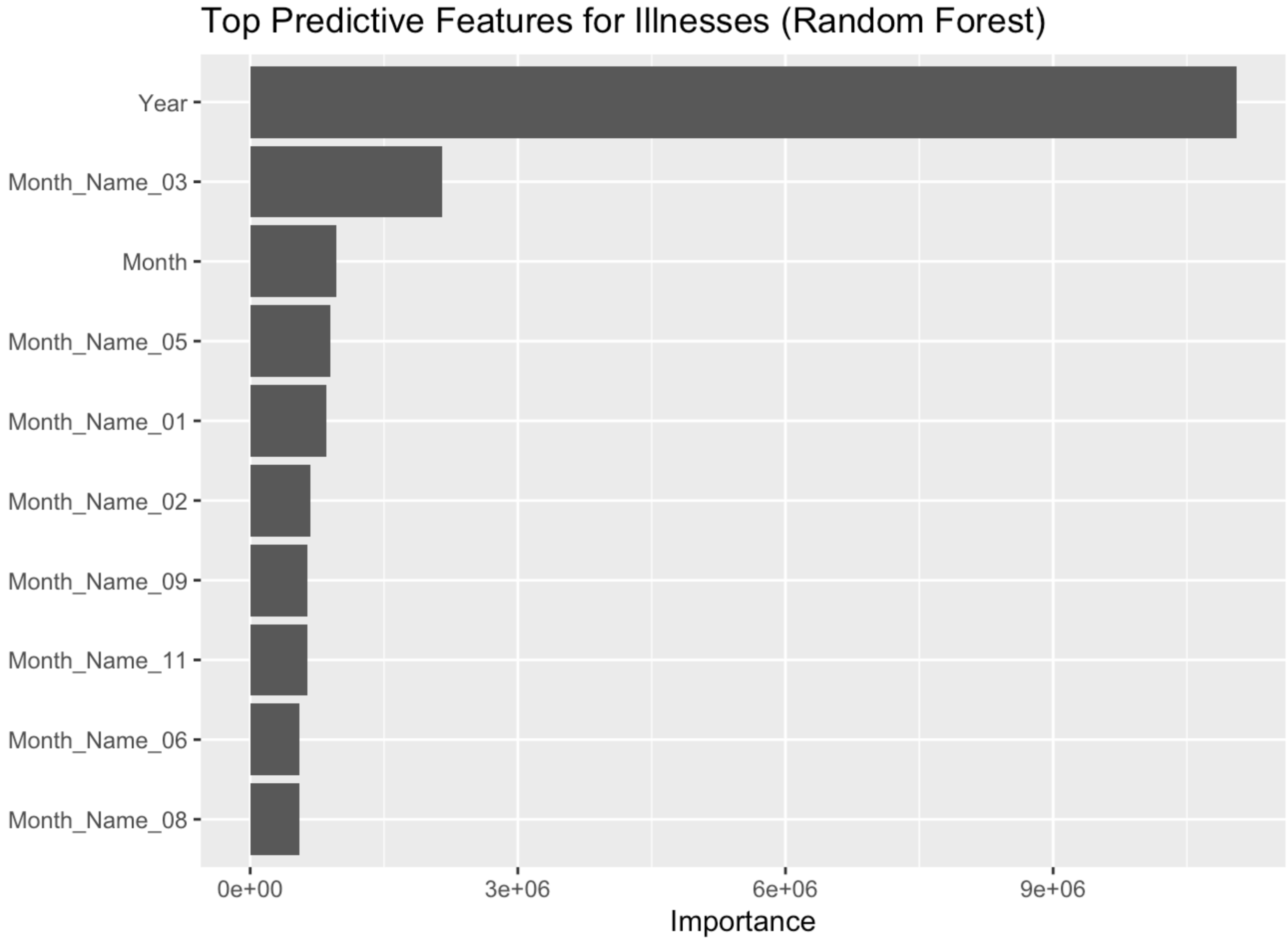
```
illness_full <- food_data_clean %>%
  group_by(Date) %>%
  summarise(Illnesses = sum(Illnesses, na.rm = TRUE)) %>%
  mutate(
    Year = year(Date),
    Month = month(Date),
    Month_Name = month(Date, label = TRUE),
    Quarter = quarter(Date),
    Season = case_when(
      Month %in% c(12, 1, 2) ~ "Winter",
      Month %in% c(3, 4, 5) ~ "Spring",
      Month %in% c(6, 7, 8) ~ "Summer",
      Month %in% c(9, 10, 11) ~ "Fall"
    )
  ) %>%
  ungroup()

illness_ml <- illness_full %>% select(-Date)
```

```
rf_spec <- rand_forest(mtry = 5, trees = 500, min_n = 10) %>%
  set_engine("randomForest") %>%
  set_mode("regression")

rec <- recipe(Illnesses ~ ., data = illness_ml) %>%
  step_impute_median(all_numeric(), -all_outcomes()) %>%
  step_dummy(all_nominal_predictors())

rf_workflow <- workflow() %>%
  add_model(rf_spec) %>%
  add_recipe(rec)

rf_fit_final <- rf_workflow %>% fit(data = illness_ml)
```

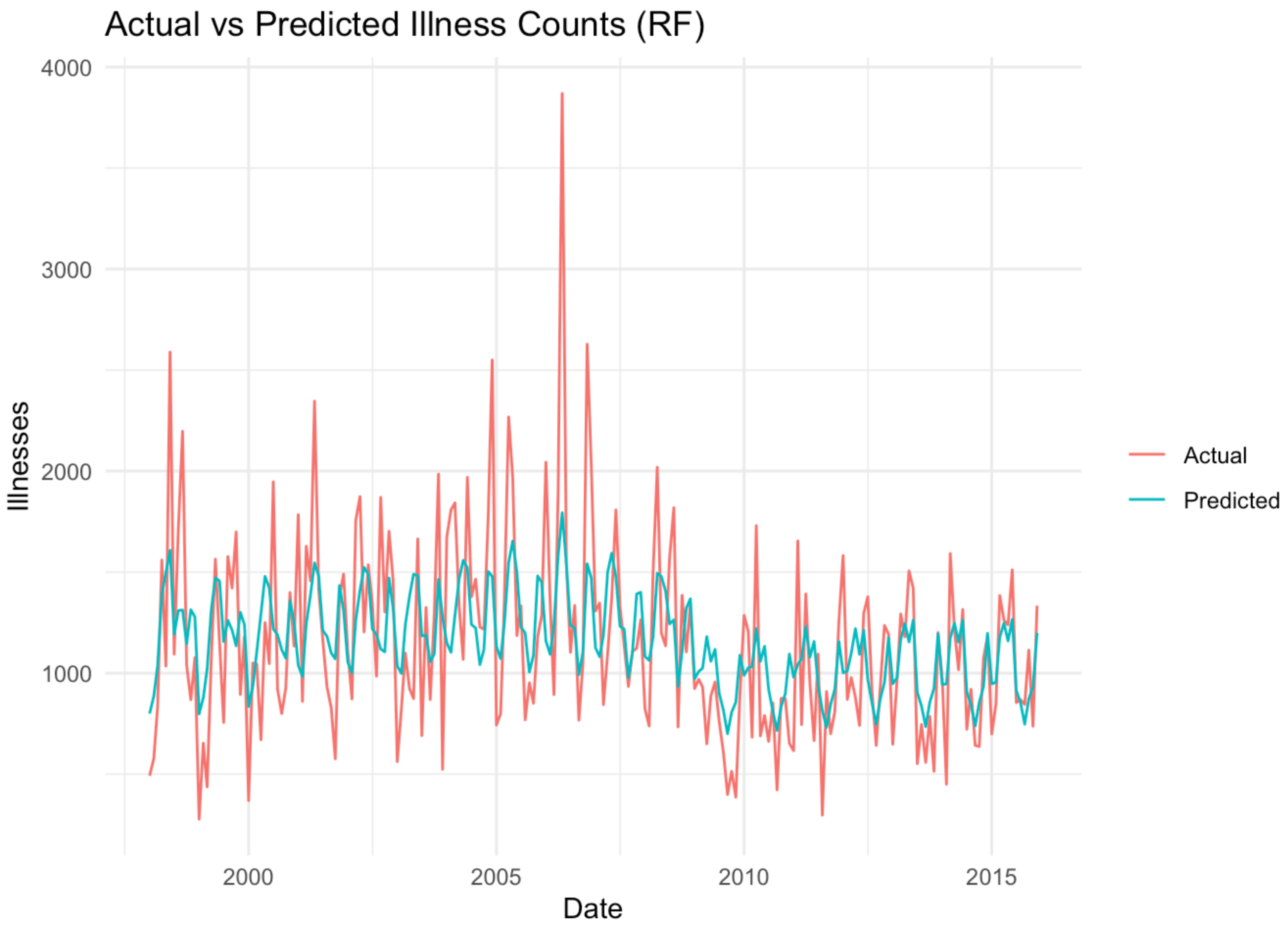
```
rf_preds <- predict(rf_fit_final, new_data = illness_ml) %>%
  bind_cols(illness_ml %>% select(Illnesses))

metrics <- rf_preds %>%
  metrics(truth = Illnesses, estimate = .pred)

print(metrics)
```

```
## # A tibble: 3 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard     370.
## 2 rsq     standard       0.462
## 3 mae     standard     280.
```

```
vip::vip(rf_fit_final$fit$fit, num_features = 10) +
  ggtitle("Top Predictive Features for Illnesses (Random Forest)")
```



```
rf_preds_plot <- predict(rf_fit_final, new_data = illness_ml) %>%
  bind_cols(illness_full %>% select(Date, Illnesses))  # ← Date comes from full version

ggplot(rf_preds_plot, aes(x = Date)) +
  geom_line(aes(y = Illnesses, color = "Actual")) +
  geom_line(aes(y = .pred, color = "Predicted")) +
  labs(title = "Actual vs Predicted Illness Counts (RF)",
       y = "Illnesses", x = "Date", color = "") +
  theme_minimal()
```



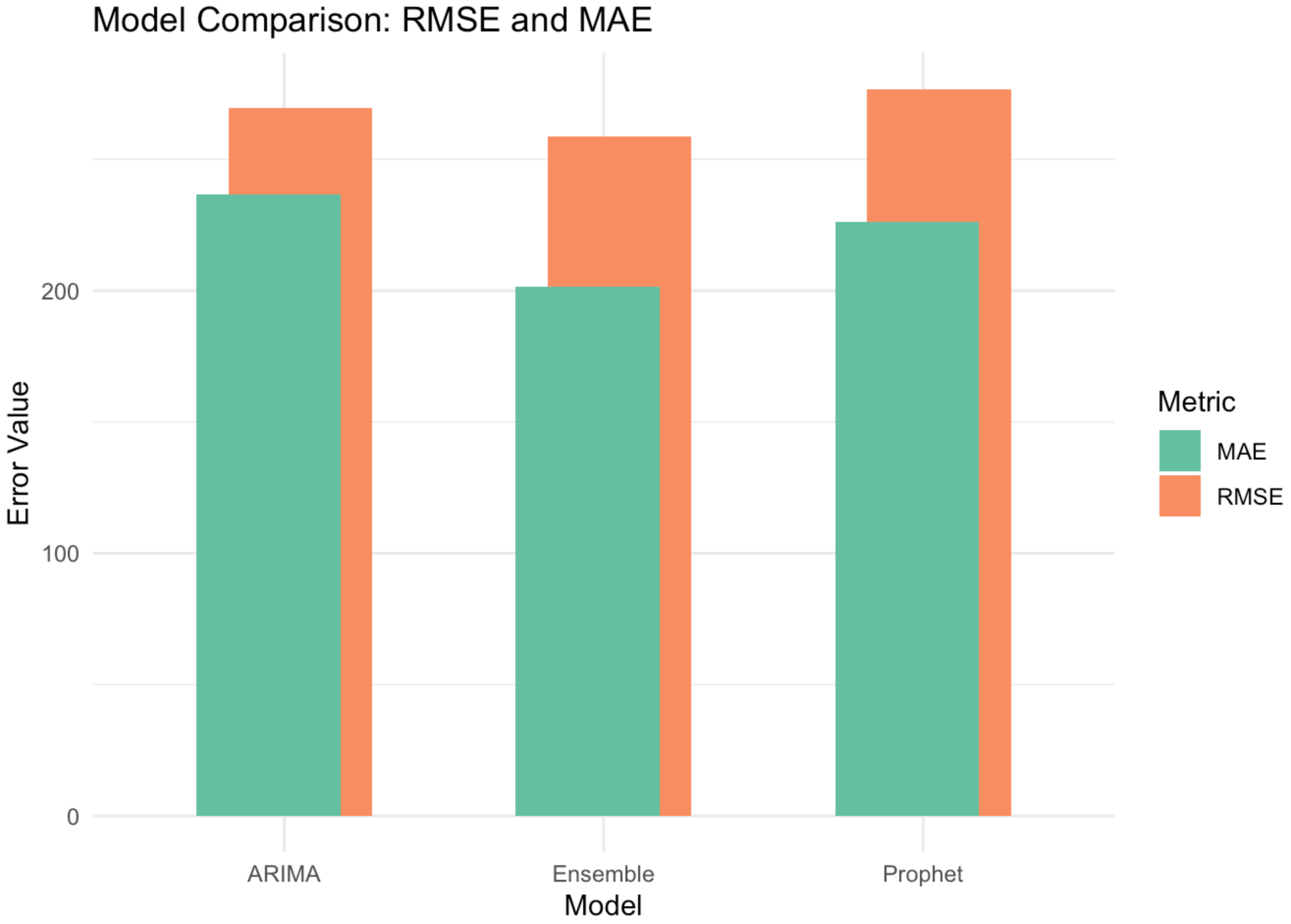
```
set.seed(42)
split <- initial_split(illness_ml, prop = 0.8)
train_data <- training(split)
test_data  <- testing(split)

rf_fit_cv <- rf_workflow %>% fit(data = train_data)

rf_test_preds <- predict(rf_fit_cv, new_data = test_data) %>%
  bind_cols(test_data %>% select(Illnesses))

metrics(rf_test_preds, truth = Illnesses, estimate = .pred)
```

```
## # A tibble: 3 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard    406.
## 2 rsq     standard      0.0680
## 3 mae     standard    334.
```

```
comparison_df <- tibble(
  truth = test_values,
  ARIMA = arima_pred,
  Prophet = prophet_pred,
  Ensemble = (arima_pred + prophet_pred) / 2
)

comparison_long <- comparison_df %>%
  tidyr::pivot_longer(-truth, names_to = "Model", values_to = ".pred")

model_metrics <- comparison_long %>%
  group_by(Model) %>%
  summarise(
    RMSE = rmse_vec(truth, .pred),
    MAE = mae_vec(truth, .pred)
  )
print(model_metrics)
```

```
## # A tibble: 3 × 3
##   Model     RMSE   MAE
##   <chr>    <dbl> <dbl>
## 1 ARIMA     269.  237.
## 2 Ensemble  259.  202.
## 3 Prophet   277.  226.
```

```
model_metrics_long <- model_metrics %>%
  pivot_longer(cols = c(RMSE, MAE), names_to = "Metric", values_to = "Value")


ggplot(model_metrics_long, aes(x = Model, y = Value, fill = Metric)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.2)) +
  labs(title = "Model Comparison: RMSE and MAE",
       x = "Model", y = "Error Value") +
  theme_minimal() +
  scale_fill_brewer(palette = "Set2")
```


Model Comparison: RMSE and MAE
Error Value
0
100
200
ARIMA
Ensemble
Prophet
Model
Metric
MAE
RMSE