Data cleaning for Exploratory Data Analysis Eastern Bluebird Capstone

Team: Yichen Le yl4347, Yuheng Shen ys2393, Linyi Xia lx277, Jia Liu jl4769, Rongjian Zhai rz495

Domain problem formulation

We aim to build a robust workflow that predicts Eastern Bluebird occurrences across their range in the eastern United States and southern Canada. The predictors appear to come from gridded environmental products summarizing land cover composition and topography. There is no accompanying metadata, so part of the project involves confirming how these variables were constructed (e.g., spatial resolution, temporal coverage, buffer size). Understanding the provenance of each variable is essential before interpreting model results or making ecological claims.

Step 1: Review background information

Information on data collection

`EasternBluebird.csv` ships with the STSCI 5954 course workspace (downloaded September 2024). Each record appears to summarize one land-cover footprint: latitude/longitude locate the grid cell, `ELEV` gives elevation, and the land-cover columns store percent cover for different habitat types. The response variable `y` marks whether Eastern Bluebirds were observed at that footprint.

No official metadata or collection notes were provided. For now we assume the values come from a remote-sensing product paired with presence/absence labels, but we still need confirmation on several points:

- Who prepared the CSV and which upstream system generated the land-cover percentages?
- Over what years (or aggregation window) were land-cover values calculated?
- Do repeated latitude/longitude pairs represent repeated visits, temporal replicates, or overlapping buffers?

When authoritative documentation becomes available, we should save a copy (PDF or local archive) and update this section with a stable link.

While we work with the interim understanding above, our cleaning objectives are:

- Determine where Eastern Bluebirds occur and are most abundant across the eastern United States (Canada to Mexico/Central America).
- Identify environmental features most associated with presence (for example, proximity to water or particular tree types).
- Summarize site-level information that can support regional and local conservation planning.

Data dictionary

The table below captures the field descriptions we currently infer from the headers. Treat these notes as provisional until we can compare them with official metadata; once an authoritative source is found, store the link (or a saved PDF) alongside this project.

```
data_dictionary <- data.frame(
  column = c(
    "LATITUDE", "LONGITUDE", "ELEV", "Shallow_Ocean", "CoastShore_lines",
```

```
    "Shallow_Inland", "Deep_Inland", "Moderate_Ocean", "Deep_Ocean",
    "Evergreen_needle", "Grasslands", "Croplands", "Urban_Built", "Barren",
    "Evergreen_broad", "Deciduous_needle", "Deciduous_broad", "Mixed_forest",
    "Closed_shrubland", "Open_shrubland", "Woody_savannas", "Savannas", "y"
  ),
  description = c(
    "Latitude of the sampling footprint in decimal degrees (WGS84).",
    "Longitude of the sampling footprint in decimal degrees (WGS84, negative for West).",
    "Elevation of the footprint in meters above sea level (negative values indicate locations below sea
    "Percent of the footprint classified as shallow ocean water.",
    "Percent of the footprint flagged as coastal shoreline interface.",
    "Percent of the footprint covered by shallow inland water bodies.",
    "Percent of the footprint covered by deep inland water bodies.",
    "Percent of the footprint in moderate-depth ocean water.",
    "Percent of the footprint in deep ocean water.",
    "Percent evergreen needleleaf forest cover.",
    "Percent grassland cover.",
    "Percent cropland or agricultural cover.",
    "Percent urban or built-up land cover.",
    "Percent barren land (bare soil/rock).",
    "Percent evergreen broadleaf forest cover.",
    "Percent deciduous needleleaf forest cover.",
    "Percent deciduous broadleaf forest cover.",
    "Percent mixed forest cover.",
    "Percent closed shrubland cover.",
    "Percent open shrubland cover.",
    "Percent woody savanna cover.",
    "Percent savanna cover.",
    "Binary indicator of Eastern Bluebird presence (1) or absence (0) for this footprint."
  ),
  stringsAsFactors = FALSE
)

knitr::kable(data_dictionary)
```

column	description
LATITUDE	Latitude of the sampling footprint in decimal degrees (WGS84).
LONGITUDE	Longitude of the sampling footprint in decimal degrees (WGS84, negative for West).
ELEV	Elevation of the footprint in meters above sea level (negative values indicate locations below sea level).
Shallow_Ocean	Percent of the footprint classified as shallow ocean water.
CoastShore_lines	Percent of the footprint flagged as coastal shoreline interface.
Shallow_Inland	Percent of the footprint covered by shallow inland water bodies.
Deep_Inland	Percent of the footprint covered by deep inland water bodies.
Moderate_Ocean	Percent of the footprint in moderate-depth ocean water.
Deep_Ocean	Percent of the footprint in deep ocean water.
Evergreen_needle	Percent evergreen needleleaf forest cover.
Grasslands	Percent grassland cover.
Croplands	Percent cropland or agricultural cover.
Urban_Built	Percent urban or built-up land cover.
Barren	Percent barren land (bare soil/rock).
Evergreen_broad	Percent evergreen broadleaf forest cover.
Deciduous_needle	Percent deciduous needleleaf forest cover.

column	description
Deciduous_broad	Percent deciduous broadleaf forest cover.
Mixed_forest	Percent mixed forest cover.
Closed_shrubland	Percent closed shrubland cover.
Open_shrubland	Percent open shrubland cover.
Woody_savannas	Percent woody savanna cover.
Savannas	Percent savanna cover.
y	Binary indicator of Eastern Bluebird presence (1) or absence (0) for this footprint.

Step 2: Load the data

We load the CSV with base R's `read.csv` and convert key columns to numeric so later steps are straightforward.

```
options(stringsAsFactors = FALSE)
```

```
data_path <- "EasternBluebird.csv"
file_details <- file.info(data_path)
file_overview <- data.frame(
  file_size_mb = round(file_details$size / 1024^2, 2),
  last_modified = file_details$mtime
)
file_overview
```

```
##   file_size_mb       last_modified
## 1         7.26 2025-10-31 15:42:33
```

```
bluebird_raw <- read.csv(data_path, stringsAsFactors = FALSE)
str(bluebird_raw)
```

```
## 'data.frame':    64724 obs. of  23 variables:
##  $ LATITUDE        : num  35.3 36 36.7 37 37.3 ...
##  $ LONGITUDE       : num  -76.6 -78.9 -81.5 -79.5 -80.5 ...
##  $ ELEV            : num  2.24 100.92 939.3 212.17 773.58 ...
##  $ Shallow_Ocean   : num  0 0 0 0 0 ...
##  $ CoastShore_lines: num  0 0 0 0 0 ...
##  $ Shallow_Inland  : num  0 0 0 0 0 ...
##  $ Deep_Inland     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Moderate_Ocean  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Deep_Ocean      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Evergreen_needle: num  40.82 0 0 2.04 0 ...
##  $ Grasslands      : num  2.04 0 0 0 0 ...
##  $ Croplands       : num  0 0 0 0 0 ...
##  $ Urban_Built     : num  0 63.9 0 0 0 ...
##  $ Barren          : num  0 0 0 0 0 ...
##  $ Evergreen_broad : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Deciduous_needle: num  0 0 0 0 0 ...
##  $ Deciduous_broad : num  0 0 100 10.2 100 ...
##  $ Mixed_forest    : num  51 11.1 0 85.7 0 ...
##  $ Closed_shrubland: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Open_shrubland  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Woody_savannas  : num  4.08 25 0 2.04 0 ...
##  $ Savannas        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ y               : int  0 0 0 0 0 0 0 0 0 0 ...
```

```r
bluebird_data <- bluebird_raw
bluebird_data$LATITUDE <- as.numeric(bluebird_data$LATITUDE)
bluebird_data$LONGITUDE <- as.numeric(bluebird_data$LONGITUDE)

landcover_cols <- c(
  "Shallow_Ocean", "CoastShore_lines", "Shallow_Inland", "Deep_Inland",
  "Moderate_Ocean", "Deep_Ocean", "Evergreen_needle", "Grasslands", "Croplands",
  "Urban_Built", "Barren", "Evergreen_broad", "Deciduous_needle",
  "Deciduous_broad", "Mixed_forest", "Closed_shrubland", "Open_shrubland",
  "Woody_savannas", "Savannas"
)

bluebird_data$landcover_total <- rowSums(bluebird_data[, landcover_cols], na.rm = TRUE)
bluebird_data$landcover_total_over_100 <- bluebird_data$landcover_total > (100 + 1e-6)

data.frame(
  rows = nrow(bluebird_data),
  columns = ncol(bluebird_data)
)
```

```
##    rows columns
## 1 64724      25
```

```r
head(bluebird_data, 5)
```

```
##   LATITUDE LONGITUDE      ELEV Shallow_Ocean CoastShore_lines Shallow_Inland
## 1 35.27266 -76.61289   2.24365             0                0              0
## 2 35.95440 -78.94340 100.91523             0                0              0
## 3 36.72264 -81.48981 939.29868             0                0              0
## 4 37.02214 -79.46737 212.17029             0                0              0
## 5 37.29057 -80.45833 773.57905             0                0              0
##   Deep_Inland Moderate_Ocean Deep_Ocean Evergreen_needle Grasslands Croplands
## 1           0              0          0        40.816327   2.040816         0
## 2           0              0          0         0.000000   0.000000         0
## 3           0              0          0         0.000000   0.000000         0
## 4           0              0          0         2.040816   0.000000         0
## 5           0              0          0         0.000000   0.000000         0
##   Urban_Built Barren Evergreen_broad Deciduous_needle Deciduous_broad
## 1     0.00000      0               0                0         0.00000
## 2    63.88889      0               0                0         0.00000
## 3     0.00000      0               0                0       100.00000
## 4     0.00000      0               0                0        10.20408
## 5     0.00000      0               0                0       100.00000
##   Mixed_forest Closed_shrubland Open_shrubland Woody_savannas Savannas y
## 1     51.02041                0              0       4.081633        0 0
## 2     11.11111                0              0      25.000000        0 0
## 3      0.00000                0              0       0.000000        0 0
## 4     85.71429                0              0       2.040816        0 0
## 5      0.00000                0              0       0.000000        0 0
##   landcover_total landcover_total_over_100
## 1        97.95918                    FALSE
## 2       100.00000                    FALSE
## 3       100.00000                    FALSE
## 4       100.00000                    FALSE
## 5       100.00000                    FALSE
```

Step 3: Examine the data

This section follows a basic checklist: look for invalid values, study missingness, confirm the table is tidy, review column names and types, and run a dataset-specific sanity check.

Invalid values

```
range_summary <- data.frame(
  lat_min = min(bluebird_data$LATITUDE, na.rm = TRUE),
  lat_max = max(bluebird_data$LATITUDE, na.rm = TRUE),
  lon_min = min(bluebird_data$LONGITUDE, na.rm = TRUE),
  lon_max = max(bluebird_data$LONGITUDE, na.rm = TRUE),
  elev_min = min(bluebird_data$ELEV, na.rm = TRUE),
  elev_max = max(bluebird_data$ELEV, na.rm = TRUE)
)
range_summary
```

```
##    lat_min  lat_max   lon_min   lon_max  elev_min elev_max
## 1 35.00075 49.98552 -84.99986 -70.00072 -17.71138 1862.057
```

Coordinates fall within the eastern United States and southern Canada, and elevations remain plausible for terrestrial sites.

Missing values

```
missing_counts <- data.frame(
  column = names(bluebird_data),
  missing_count = colSums(is.na(bluebird_data))
)
missing_counts
```

```
##                                        column missing_count
## LATITUDE                             LATITUDE             0
## LONGITUDE                           LONGITUDE             0
## ELEV                                     ELEV             0
## Shallow_Ocean                     Shallow_Ocean             0
## CoastShore_lines                CoastShore_lines             0
## Shallow_Inland                    Shallow_Inland             0
## Deep_Inland                          Deep_Inland             0
## Moderate_Ocean                    Moderate_Ocean             0
## Deep_Ocean                           Deep_Ocean             0
## Evergreen_needle                Evergreen_needle             0
## Grasslands                            Grasslands             0
## Croplands                              Croplands             0
## Urban_Built                          Urban_Built             0
## Barren                                    Barren             0
## Evergreen_broad                  Evergreen_broad             0
## Deciduous_needle                Deciduous_needle             0
## Deciduous_broad                  Deciduous_broad             0
## Mixed_forest                        Mixed_forest             0
## Closed_shrubland                Closed_shrubland             0
## Open_shrubland                    Open_shrubland             0
## Woody_savannas                    Woody_savannas             0
## Savannas                                Savannas             0
## y                                               y             0
```

```
## landcover_total                   landcover_total             0
## landcover_total_over_100 landcover_total_over_100             0
```

```
missing_rows <- bluebird_data[!complete.cases(bluebird_data), ]
missing_rows
```

```
##  [1] LATITUDE                LONGITUDE               ELEV
##  [4] Shallow_Ocean           CoastShore_lines        Shallow_Inland
##  [7] Deep_Inland             Moderate_Ocean          Deep_Ocean
## [10] Evergreen_needle        Grasslands              Croplands
## [13] Urban_Built             Barren                  Evergreen_broad
## [16] Deciduous_needle        Deciduous_broad         Mixed_forest
## [19] Closed_shrubland        Open_shrubland          Woody_savannas
## [22] Savannas                y                       landcover_total
## [25] landcover_total_over_100
## <0 rows> (or 0-length row.names)
```

Missing values would be recorded as `NA`. Every column reports zero `NA`s and no incomplete rows appear, so downstream analyses can proceed without imputation or special-case handling.

Data format

```
site_keys <- paste(bluebird_data$LATITUDE, bluebird_data$LONGITUDE)
site_table <- table(site_keys)
total_sites <- length(site_table)
max_records <- max(site_table)
sites_with_duplicates <- sum(site_table > 1)
pct_sites_with_duplicates <- round((sites_with_duplicates / total_sites) * 100, 2)
duplicate_summary <- data.frame(
  total_sites = total_sites,
  max_records = max_records,
  sites_with_duplicates = sites_with_duplicates,
  pct_sites_with_duplicates = paste0(pct_sites_with_duplicates, "%")
)
duplicate_summary
```

```
##   total_sites max_records sites_with_duplicates pct_sites_with_duplicates
## 1       36434          51                  9296                     25.51%
```

Each row already represents one survey footprint (an observational unit), and columns hold single measurements. Repeated latitude/longitude pairs indicate multiple surveys at the same site, so any train/test split should keep those replicate rows together.

Column names

```
data.frame(column = names(bluebird_data))
```

```
##                      column
## 1                  LATITUDE
## 2                 LONGITUDE
## 3                      ELEV
## 4             Shallow_Ocean
## 5          CoastShore_lines
## 6            Shallow_Inland
## 7               Deep_Inland
## 8            Moderate_Ocean
```

```
## 9                Deep_Ocean
## 10         Evergreen_needle
## 11               Grasslands
## 12                Croplands
## 13              Urban_Built
## 14                   Barren
## 15          Evergreen_broad
## 16         Deciduous_needle
## 17          Deciduous_broad
## 18             Mixed_forest
## 19         Closed_shrubland
## 20           Open_shrubland
## 21           Woody_savannas
## 22                 Savannas
## 23                        y
## 24          landcover_total
## 25 landcover_total_over_100
```

Column names already use underscores, so we keep them as provided.

Variable type

```
variable_types <- data.frame(
  column = names(bluebird_data),
  class = sapply(bluebird_data, function(x) paste(class(x), collapse = ", "))
)
variable_types
```

```
##                                         column   class
## LATITUDE                               LATITUDE numeric
## LONGITUDE                             LONGITUDE numeric
## ELEV                                       ELEV numeric
## Shallow_Ocean                     Shallow_Ocean numeric
## CoastShore_lines                CoastShore_lines numeric
## Shallow_Inland                    Shallow_Inland numeric
## Deep_Inland                          Deep_Inland integer
## Moderate_Ocean                  Moderate_Ocean numeric
## Deep_Ocean                          Deep_Ocean integer
## Evergreen_needle                Evergreen_needle numeric
## Grasslands                            Grasslands numeric
## Croplands                              Croplands numeric
## Urban_Built                          Urban_Built numeric
## Barren                                     Barren numeric
## Evergreen_broad                  Evergreen_broad numeric
## Deciduous_needle                Deciduous_needle numeric
## Deciduous_broad                  Deciduous_broad numeric
## Mixed_forest                        Mixed_forest numeric
## Closed_shrubland                Closed_shrubland numeric
## Open_shrubland                    Open_shrubland numeric
## Woody_savannas                    Woody_savannas numeric
## Savannas                                Savannas numeric
## y                                               y integer
## landcover_total                  landcover_total numeric
## landcover_total_over_100 landcover_total_over_100 logical
```

Predictors are numeric, and the response `y` remains coded as 0/1. If factor semantics are required, the conversion can occur closer to modeling.

Data specific explorations

```
landcover_stats <- data.frame(
  min_total = min(bluebird_data$landcover_total, na.rm = TRUE),
  p10_total = as.numeric(quantile(bluebird_data$landcover_total, 0.10, na.rm = TRUE)),
  median_total = median(bluebird_data$landcover_total, na.rm = TRUE),
  mean_total = mean(bluebird_data$landcover_total, na.rm = TRUE),
  p90_total = as.numeric(quantile(bluebird_data$landcover_total, 0.90, na.rm = TRUE)),
  max_total = max(bluebird_data$landcover_total, na.rm = TRUE),
  pct_over_100 = paste0(round(mean(bluebird_data$landcover_total_over_100, na.rm = TRUE) * 100, 2), "%")
)
landcover_stats
```

```
##   min_total p10_total median_total mean_total p90_total max_total pct_over_100
## 1   54.7619       100          100   105.9745  130.5556       200       18.92%
```

Land cover totals cluster around 100 but occasionally exceed it, reinforcing the need to confirm whether overlapping buffers or stacked categories generated the export.

Step 4: Clean the data

The cleaning workflow below:

1. Copies the raw table and stores latitude/longitude as numeric values.
2. Computes land cover totals and flags rows where totals exceed 100.
3. Adds normalized land cover fractions so each record sums to one even if the original totals differ.
4. Aggregates to a site-level table (one row per latitude/longitude) with counts of replicate observations and simple presence summaries. This reduces leakage when splitting data by location.

```
bluebird_simple <- bluebird_data

for (col in landcover_cols) {
  new_name <- paste0(col, "_frac")
  bluebird_simple[[new_name]] <- ifelse(
    bluebird_simple$landcover_total == 0,
    0,
    bluebird_simple[[col]] / bluebird_simple$landcover_total
  )
}

site_counts <- aggregate(y ~ LATITUDE + LONGITUDE, data = bluebird_simple, length)
names(site_counts)[3] <- "n_observations"

presence_any <- aggregate(y ~ LATITUDE + LONGITUDE, data = bluebird_simple, function(x) as.integer(any(
names(presence_any)[3] <- "presence_any"

presence_rate <- aggregate(y ~ LATITUDE + LONGITUDE, data = bluebird_simple, mean)
names(presence_rate)[3] <- "presence_rate"

elev_mean <- aggregate(ELEV ~ LATITUDE + LONGITUDE, data = bluebird_simple, mean)
names(elev_mean)[3] <- "elev_mean"

landcover_total_mean <- aggregate(landcover_total ~ LATITUDE + LONGITUDE, data = bluebird_simple, mean)
```

```r
names(landcover_total_mean)[3] <- "landcover_total_mean"

site_level <- merge(site_counts, presence_any, by = c("LATITUDE", "LONGITUDE"))
site_level <- merge(site_level, presence_rate, by = c("LATITUDE", "LONGITUDE"))
site_level <- merge(site_level, elev_mean, by = c("LATITUDE", "LONGITUDE"))
site_level <- merge(site_level, landcover_total_mean, by = c("LATITUDE", "LONGITUDE"))

bluebird_clean_list <- list(
  cleaned = bluebird_simple,
  site_level = site_level
)

site_overview <- data.frame(
  sites = nrow(site_level),
  mean_records_per_site = mean(site_level$n_observations),
  max_records_per_site = max(site_level$n_observations)
)
site_overview
```

```
##   sites mean_records_per_site max_records_per_site
## 1 36434              1.776473                   51
```

```r
columns_to_show <- c(
  "LATITUDE", "LONGITUDE", "ELEV", "landcover_total",
  "landcover_total_over_100", "Woody_savannas", "Woody_savannas_frac", "y"
)
head(bluebird_clean_list$cleaned[, columns_to_show], 5)
```

```
##   LATITUDE LONGITUDE      ELEV landcover_total landcover_total_over_100
## 1 35.27266 -76.61289   2.24365        97.95918                     FALSE
## 2 35.95440 -78.94340 100.91523       100.00000                     FALSE
## 3 36.72264 -81.48981 939.29868       100.00000                     FALSE
## 4 37.02214 -79.46737 212.17029       100.00000                     FALSE
## 5 37.29057 -80.45833 773.57905       100.00000                     FALSE
##   Woody_savannas Woody_savannas_frac y
## 1       4.081633          0.04166667 0
## 2      25.000000          0.25000000 0
## 3       0.000000          0.00000000 0
## 4       2.040816          0.02040816 0
## 5       0.000000          0.00000000 0
```

```r
head(bluebird_clean_list$site_level, 5)
```

```
##   LATITUDE LONGITUDE n_observations presence_any presence_rate  elev_mean
## 1 35.00075 -80.63416              1            0             0  204.21820
## 2 35.00108 -79.06775              1            0             0   69.19665
## 3 35.00395 -83.17117              1            0             0  794.37719
## 4 35.00684 -80.63347              1            0             0  198.29341
## 5 35.00952 -83.32141              1            0             0 1077.04045
##   landcover_total_mean
## 1                  100
## 2                  100
## 3                  100
## 4                  100
## 5                  100
```

```r
site_stats <- data.frame(
  min_rate = min(bluebird_clean_list$site_level$presence_rate, na.rm = TRUE),
  mean_rate = mean(bluebird_clean_list$site_level$presence_rate, na.rm = TRUE),
  median_rate = median(bluebird_clean_list$site_level$presence_rate, na.rm = TRUE),
  max_rate = max(bluebird_clean_list$site_level$presence_rate, na.rm = TRUE)
)
site_stats
```

```
##   min_rate  mean_rate median_rate max_rate
## 1        0 0.05617417           0        1
```

Next steps:

- Confirm the interpretation of land cover totals that exceed 100 and determine whether they require re-normalization or stratified handling.
- Decide on a final export format (CSV, parquet, or RDS) for both the observation-level and site-level tables once metadata questions are resolved.
- Incorporate temporal information if it is available elsewhere so we can respect survey-years during modeling splits.