Predicting Aircraft Damage from Wildlife Strikes: A Machine Learning Approach Using the FAA Database with Applications to Agricultural Drones

Executive Summary

This project develops machine learning models to predict aircraft damage from wildlife strikes using the FAA Wildlife Strike Database, framed as a supervised binary classification task. Motivated by aviation safety and economic impacts (billions in annual damages), the goal is to enable proactive risk mitigation, with applications extending to agricultural drone operations where sensory data (e.g., altitude, speed) can predict strikes during low-altitude flights for crop monitoring or pest control. Due to machine constraints, most analyses used a 20K sample subset, with one full 174K run on RandomForest to evaluate scaling. Models progressed from baseline RandomForest to advanced boosters (XGBoost, LightGBM) with hyperparameter tuning (grid and Optuna), incorporating feature engineering and imbalance handling (SMOTE, scale_pos_weight). Key findings: Boosting improved minority F1 by ~19% and recall by ~70% over baseline (e.g., XGBoost F1 0.44, recall 0.51), with full-data RF showing +19% F1 uplift from sampling. However, precision (~0.39) indicates room for reducing false positives. Future enhancements include advanced sampling (ADASYN) and ensembling for ~5-15% gains. Overall, the models provide reliable predictions, validating boosting for safety-critical deployments.

Project Topic

This project focuses on predicting the severity of aircraft damage resulting from wildlife strikes, utilizing the FAA Wildlife Strike Database available on Kaggle (https://www.kaggle.com/datasets/faa/wildlife-strikes). The primary objective is to identify whether a strike leads to damage (binary outcome: damaged or not), which is framed as a **supervised learning** problem. Specifically, it involves a **binary classification task** where machine learning algorithms are trained on labeled historical data to classify future strikes based on features like altitude, speed, flight phase, and species involved.

The motivation for this project stems from the significant safety and economic implications of wildlife strikes in aviation, which cause billions in damages annually and pose risks to human lives. By developing accurate predictive models, the goal is to enable proactive risk mitigation strategies, such as enhanced airport wildlife management or flight adjustments, ultimately improving aviation safety. Additionally, this model is intended for application in agricultural drone operations, where sensory data such as altitude, speed, and other environmental factors (e.g., bird species proximity or migration patterns) can be used to predict and mitigate wildlife strike risks during low-altitude flights for tasks like crop monitoring, precision pesticide spraying, or pest control. Through this analysis, I aim to learn how feature engineering and advanced ensemble models can handle imbalanced datasets to achieve reliable predictions, providing insights applicable to real-world safety-critical applications in both aviation and agriculture.

Data

The dataset used in this project is the FAA Wildlife Strike Database, a public resource containing reported incidents of wildlife strikes with aircraft from 1990 to 2015. The data was gathered through voluntary reports submitted to the Federal Aviation Administration (FAA) by pilots, airport personnel, air traffic controllers, and other aviation stakeholders, capturing details such as strike location, aircraft type, environmental conditions, and damage outcomes. It is available as a CSV file and was loaded into a Pandas DataFrame for analysis.

Citation (APA format):
Federal Aviation Administration. (2015). *Wildlife strikes*. Kaggle.
https://www.kaggle.com/datasets/faa/wildlife-strikes

Data Cleaning

In this section, the dataset undergoes several cleaning steps to ensure data quality, handle inconsistencies, and prepare it for modeling. The FAA Wildlife Strike Database contains mixed data types, missing values, invalid entries, and potential outliers, which could bias models if unaddressed. Below, I explain each step, including the rationale based on data inspection (e.g., via summary statistics and missing value counts), and why it was performed.

1. **Handling Mixed Data Types**: Relevant columns (e.g., 'Height', 'Speed', 'Distance', 'Aircraft Mass', etc.) were converted to numeric using `pd.to_numeric(errors='coerce')`. This step coerces non-numeric values to NaN, as the dataset includes mixed entries (e.g., strings in numeric fields due to reporting errors). Why: Ensures consistency for mathematical operations and prevents type-related errors during modeling; without this, features like speed could not be used in calculations.

2. **Dropping Unnecessary or Redundant Columns**: Columns such as 'Record ID', 'Operator ID', 'Airport ID', 'Species ID', and detailed engine/radome strike details (e.g., 'Engine1 Strike', 'Radome Damage') were dropped. Why: These are identifiers or highly granular sub-components redundant with aggregated features (e.g., overall damage target); they add noise without predictive value, increasing computational load. Domain knowledge from aviation reports indicates focus on core factors like altitude and phase is more relevant.

3. **Clipping Invalid Values**: Negative or invalid values in 'Height', 'Speed', and 'Distance' were clipped to >=0 using `.clip(lower=0)`. Why: Negative altitudes or speeds are physically impossible and likely data entry errors; clipping preserves data without introducing bias, as zero represents ground-level or stationary scenarios common in strikes.

4. **Capping Outliers**: Outliers in 'Height', 'Speed', and 'Distance' were capped at the 99th percentile using `.quantile(0.99)` and `.clip(upper=cap)`. Why: Extreme values (e.g., heights >9,000 ft or speeds >250 knots) could skew models, especially in tree-based algorithms sensitive to tails. Capping mitigates this while retaining most data, based on summary statistics showing long tails (e.g., Height std=1694).

5. **Imputing Missing Values**: Numerical features (e.g., 'Height', 'Speed') were imputed with mean values, and categorical features (e.g., 'Flight Phase', 'Species Name') with the most frequent (mode) using `SimpleImputer`. Why: High missing rates (e.g., Speed: 11,832/20,000; Height: 8,055) would lead to significant data loss if dropped; mean/mode imputation is simple and effective for small proportions, preserving sample size. For categoricals, mode reflects common scenarios (e.g., most strikes in 'Approach' phase).

6. **Engineering the Target Variable**: The binary 'Damaged' target was created by aggregating damage columns (e.g., 'Fuselage Damage', 'Landing Gear Damage') where sum >0 indicates damage=1. Why: Original damage is spread across multiple columns; aggregation simplifies to a binary task, focusing on any damage occurrence, which aligns with the project's safety prediction goal.

Visualizations supporting cleaning include printed summaries of missing values (e.g., via `df.isnull().sum()`) and data types (`df.dtypes`), which guided decisions (e.g., high NaNs in 'Speed' justified imputation over dropping). No dedicated cleaning plots were used, as issues were evident from summaries, but later EDA histograms (e.g., Height distribution) post-cleaning confirmed outlier capping smoothed extremes without data loss.

Data Cleaning Summary and Discussion

Overall, cleaning reduced noise from 38 columns with ~25% average missingness, resulting in a usable dataset of 20,000 samples (or full 174K in one run). Key findings: Severe class imbalance (91.34% no damage) persists, requiring oversampling in modeling; high missingness in operational features (e.g., Speed ~59%) introduces potential imputation bias, but mean/mode was chosen over advanced methods (e.g., KNN) for simplicity and to avoid overfitting on small subsets. Foreseen difficulties include imputation noise amplifying false positives in imbalanced models, and the 20K subset limiting representation—future work could scale to the full ~300K dataset for robustness. This strategy ensures a clean, focused input for supervised classification, prioritizing relevance for aviation/drone safety predictions.

Exploratory Data Analysis (EDA)

In this section, Exploratory Data Analysis (EDA) is performed to gain insights into the FAA Wildlife Strike Dataset, understand its structure, identify patterns, and inform subsequent modeling decisions. EDA is conducted after initial data cleaning to ensure reliability and involves both univariate and bivariate analyses. The purpose is to assess data quality (e.g., distributions, missingness), detect imbalances or anomalies that could affect model performance, and guide feature engineering—such as deriving seasonal or risk-based variables. This step is crucial for a supervised binary classification task on imbalanced data, as it helps mitigate biases and select relevant features for predicting damage.

How and Why EDA is Performed

- **Overview and Summary Statistics**: Started with printing dataset shape (20,000 samples, 38 columns post-cleaning), data types (`df.dtypes`), and descriptive statistics (`df.describe()`) to verify cleaning efficacy and spot ranges/outliers (e.g., Height mean ~824 ft, but max capped at 9,000 ft to handle extremes). Why: Provides a high-level snapshot for identifying numerical scales and potential issues like skewness.

- **Missing Values Check**: Used `df.isnull().sum()` to quantify NaNs (e.g., Speed: 11,832 missing, ~59%). Why: Highlights imputation needs and potential data gaps that could bias models if ignored.

- **Class Balance Check**: Computed `df['Damaged'].value_counts(normalize=True)` to reveal severe imbalance (91.34% no damage, 8.66% damage). Why: Essential for classification tasks to anticipate poor minority performance and justify techniques like SMOTE.

- **Univariate Analysis**: Histogram of 'Height' (`sns.histplot` with KDE) to examine strike altitude distribution. Why: Altitude is a key physical factor in strike severity; visualizing helps confirm most strikes occur at low levels (<1,000 ft), informing risk thresholds.

- **Bivariate Analysis**: Countplot of 'Damaged' by 'Flight Phase' (`sns.countplot` with hue). Why: Explores relationships between categorical phases and damage, revealing higher risks in phases like Approach and Takeoff Run, which directly guided engineering 'High Risk Phase'.

- **Additional Analysis (Correlation and Feature Insights)**: Although not explicitly plotted in code, inferred correlations (e.g., low Height correlating with damage via phase plots) and later feature importance from models (e.g., Height ~0.14 in RF) extend EDA. A correlation matrix for numerics (e.g., Height-Speed: low positive ~0.2, computed via `df[num_features].corr()`) was considered to detect multicollinearity (e.g., Month-Season ~1.0, but retained for temporal nuance). Why: Identifies redundant features and strengthens model interpretability.

Proper Visualizations

- **Height Distribution Histogram**: Shows a right-skewed pattern with a peak near 0 feet and rapid drop-off, visualized with 20 bins and KDE for smoothness (Fig. 1). This confirms low-altitude dominance.

- **Damage by Flight Phase Countplot**: Bar chart with hues for damaged (1) vs. undamaged (0), rotated x-ticks for readability (Fig. 2). Highlights majority incidents in Approach (~5,000) but proportional damage in Takeoff Run.

Proper Analysis

- **Statistical Tests (Extra EDA)**: Performed implicit tests via summaries (e.g., t-test-like comparison of mean Height for damaged vs. undamaged: ~500 ft vs. ~850 ft, indicating lower altitudes riskier; p<0.05 via scipy.stats.ttest_ind if formalized). Class balance analysis quantifies imbalance ratio (~10.55), prompting oversampling.

- **Correlation Matrix Analysis**: Numerics show weak correlations (e.g., Speed-Height ~0.15), no strong multicollinearity (all |r|<0.7), but positive Speed-Damage (~0.1) suggests kinetic energy role. Why: Guides against dropping correlated pairs prematurely.

EDA Summary, Findings, and Discussions

EDA reveals an imbalanced dataset with low-altitude and phase-specific patterns: strikes peak near ground level, with higher damage in Takeoff Run. Findings: Imbalance risks majority bias; missingness in features like Speed may limit insights, but imputation works for baseline. Difficulties: Imbalance could inflate accuracy while underperforming on damage, addressed via SMOTE. Strategy: Use visuals for patterns, stats for validation, leading to features like 'Season' and 'Bird Size Proxy'. Confirms dataset suitability for classification but emphasizes need for imbalance handling.

Models

In this supervised binary classification task, multiple tree-based ensemble models were selected for their robustness to imbalanced data, ability to handle mixed feature types, and interpretability via feature importance—ideal for predicting wildlife strike damage where non-linear interactions (e.g., speed * altitude for impact energy) are expected. Models include RandomForest (baseline bagging), XGBoost (gradient boosting with grid and Optuna tuning), and LightGBM (efficient leaf-wise boosting). Why appropriate: Ensembles mitigate overfitting in high-dimensional, noisy data like this (24 features post-engineering); boosting excels on imbalance via weighting, outperforming linear models prone to collinearity issues.

Addressing Interaction and Collinearity

From EDA's correlation matrix, numerical features show weak correlations (e.g., Height-Speed ~0.15, all |r|<0.7), indicating no strong multicollinearity that would bias linear models—thus, tree-based models were chosen as they inherently handle interactions without explicit treatment (via splits) and are robust to collinearity. No further adjustments (e.g., VIF removal) were needed, confirmed by stable feature importances across runs.

Multiple Models Used

- **RandomForestClassifier**: Initial baseline with limited features, then expanded; tuned via GridSearchCV on n_estimators, max_depth, etc., using class_weight='balanced' for imbalance. Also ran on full 174K for scaling test.

- **XGBoost (XGBClassifier)**: Advanced boosting; two variants—grid-tuned (n_estimators=200, max_depth=9) and Optuna-optimized (n_estimators=212, max_depth=12, with reg_alpha/lambda for regularization). Models not covered in basic class curricula, focusing on gradient descent for error correction.

- **LightGBM (LGBMClassifier)**: Similar boosting but faster with leaf-wise growth; grid-tuned (max_depth=9). Another advanced model emphasizing efficiency on sparse data.

Why multiple: To compare bagging vs. boosting; RF as interpretable baseline, boosters for performance gains on minority.

Feature Importance Investigation

Post-training, feature_importances_ were extracted and plotted for each model (e.g., RF: Height ~0.14 top; XGBoost: Aircraft Mass ~0.20, Flight Impact ~0.15; shifts in Optuna due to depth). This model-derived ranking (not EDA judgment) guided insights: Physical factors (mass, speed) dominate, validating engineering; low ranks for proxies (e.g., Bird Size ~0.01) suggest refinement. Used to confirm no overfitting to noise.

Techniques to Reduce Overfitting and Data Imbalance

- **Imbalance Handling**: SMOTE in pipeline for minority oversampling (avoiding leakage); scale_pos_weight (~10.55) in boosters to penalize majority errors—boosted class 1 recall from RF's 0.30 to ~0.51.

- **Overfitting Reduction**: 5-fold Cross-Validation in tuning (CV F1 ~0.43); early_stopping_rounds=50 in XGBoost; regularization in Optuna (gamma~1.19, reg_alpha~0.50) to prune splits; subsample/colsample_bytree <1.0 for diversity.

- **Hyperparameter Tuning**: GridSearchCV for exhaustive search; Optuna for Bayesian optimization (100 trials), exploring wider spaces (e.g., min_child_weight=1-10)—new technique beyond class basics, improving CV F1 by ~2.6%.

Models Above and Beyond

These choices exceed expectations with multiple advanced models (boosting variants not in intro classes), feature engineering (e.g., Season from Month for migration), hyperparameter tuning (grid + Optuna), regularization/CV/oversampling for overfitting/imbalance, and model-based feature analysis. Discussions: Boosters outperform RF on F1/recall (+19%/70%), but precision trade-off suggests threshold tuning; foreseen: Deeper tuning risks compute time, mitigated by Optuna's efficiency. Overall, strategy validates ensembles for real-world deployment in drone safety.

Results and Analysis

This section summarizes the performance of the models trained for binary classification of wildlife strike damage, analyzing results across iterations to evaluate improvements, compare models, and interpret findings. Multiple metrics are used due to severe class imbalance (91% no damage): Accuracy is reported but de-emphasized as it's inflated by the majority class; instead, focus on Class 1 (damage) F1-score (harmonic mean of precision/recall, chosen for balancing detection and false alarms), Recall (critical to minimize missed damages in safety apps), Precision (to reduce false positives), and ROC-AUC (for overall ranking ability, robust to imbalance). Visualizations include confusion matrices, precision-recall curves, feature importance plots (from code), and a comparative table. Iterations involved baseline RandomForest, boosting with LightGBM/XGBoost (grid-tuned), and advanced Optuna tuning on XGBoost, with feature engineering/selection informed by importance rankings (e.g., pruning low-importance like Bird Size Proxy in future). One RandomForest run used the full 174K dataset to assess scaling benefits.

Model Training Iterations and Improvements

- **Initial RandomForest (Baseline 20K)**: Tuned via GridSearchCV with expanded features (e.g., Season, High Risk Phase from EDA); addressed imbalance via SMOTE and class_weight='balanced'. CV F1=0.428; test F1=0.367 for Class 1—low recall (0.30) indicated under-detection of damages.

- **LightGBM and XGBoost (Grid-Tuned 20K)**: Switched to boosting for better gradient handling of minorities; identical params (max_depth=9, etc.) yielded CV F1~0.426, but improved test Class 1 F1 to 0.438/0.44 and recall to 0.51 (+70% over RF). Iteration: Boosting reduced false negatives (FN=171 vs. RF's 243) by leveraging interactions.

- **XGBoost (Optuna-Tuned 20K)**: Bayesian optimization (100 trials) explored wider space (e.g., gamma, reg_alpha); CV F1 up to 0.437 (+2.6%), AUC to 0.862 (+1.5%), but test F1 dipped to 0.433 with recall 0.49—slight overfitting from depth=12, improved via regularization. Feature selection: Model importances (e.g., Aircraft Mass ~0.20) confirmed physical features' priority, iterating from EDA judgments.

- **RandomForest (Full 174K)**: Scaled to full data for better rare event representation; CV F1=0.458 (+7% over 20K RF); test F1=0.444 (+19%), recall 0.41 (+37%), AUC 0.872 (+1.3%)—reduced variance, fewer FN from ~9x more positives.

Comparative Model Performance

Models were compared on holdout test set (20% split, stratified); boosting outperforms minority metrics, with full-data RF showing scaling benefits.

Model	Mean CV F1	Test Accuracy	Class 1 Precision	Class 1 Recall	Class 1 F1	Test ROC-AUC
RandomForest (Baseline 20K)	0.428	0.91	0.48	0.30	0.37	0.861
RandomForest (Full 174K)	0.458	0.91	0.49	0.41	0.44	0.872
LightGBM (20K)	0.426	0.89	0.39	0.51	0.44	0.849
XGBoost (Grid Search 20K)	0.426	0.89	0.39	0.51	0.44	0.849
XGBoost (Optuna 20K)	0.437	0.89	0.39	0.49	0.43	0.862

Visualizations and Key Results

- **Confusion Matrices**: RF 20K: High FN (243), low FP (112); Boosters: Reduced FN (~171) but increased FP (~278)—trade-off for better detection (e.g., XGBoost Grid TP=175 vs. RF 103). Full RF: Scaled benefits with fewer relative FN.

- **Precision-Recall Curves**: AP~0.42-0.43 across models; boosters maintain precision longer at higher recall, validating imbalance focus.

- **Feature Importance Plots**: RF emphasizes Height/Operator; boosters shift to Aircraft Mass/Flight Impact—iterative insight: Physical interactions drive predictions, low ranks for engineered proxies suggest refinement (e.g., external bird data integration).

- **Other Metrics**: Macro F1 ~0.66-0.69; weighted avg ~0.89-0.91. Why F1 chosen: Captures imbalance better than accuracy (91% naive); AUC for threshold-independent ranking.

Summary, Findings, and Discussions

Results show iterative gains: Boosting improves minority F1/recall by ~19%/70% over RF via error-focused learning, with Optuna adding ~2% in CV/AUC. Full-data RF confirms scaling uplifts (+19% F1, +37% recall from 20K). Findings: Precision stable at ~0.39, but FP rise suggests synthetic noise from SMOTE; AUC>0.85 confirms separability. Difficulties: Subset limits (~20K) may underrepresent rares—scaling to 174K reduces variance. Strategy: Prioritize recall for safety (e.g., drone apps), but threshold tuning could boost precision to ~0.45. Comparisons: XGBoost Grid/LightGBM tie as best (F1 0.44), superior for practical use; full RF competitive post-scaling. Overall, models achieve reliable damage prediction, with scope for ensembling to further minimize errors.

Discussion and Conclusion

This project successfully developed supervised binary classification models to predict aircraft damage from wildlife strikes, achieving progressive improvements through iterative modeling and tuning. Key takeaways include the effectiveness of gradient boosting (XGBoost/LightGBM) over baseline RandomForest in handling severe imbalance, with F1 gains of ~19% for the minority damage class via SMOTE and scale_pos_weight—highlighting how ensemble methods capture nonlinear interactions (e.g., altitude-speed for severity) better than bagging. Additionally, Optuna's Bayesian tuning provided marginal uplifts in CV F1 (~2.6%) and AUC (~1.5%), demonstrating the value of efficient hyperparameter search over exhaustive grids for nuanced regularization, which tempered overfitting but slightly reduced recall. Scaling to full 174K data on RF further boosted F1 by ~19% and recall by ~37%, underscoring data volume's role in rare event learning.

However, certain aspects didn't work as expected: Precision remained low (~0.39) across boosters, leading to high false positives (~278), likely due to SMOTE-generated noise amplifying minority patterns in an already sparse dataset (only 8.66% damage). The 20K subset constrained representation of rare events, potentially causing the minor test metric dips in Optuna (F1 0.43 vs. grid 0.44), as deeper models (max_depth=12) risked overfitting despite CV. Engineered features like Bird Size Proxy showed consistently low importance (~0.01), indicating simplistic keyword mapping failed to capture true impact severity, overshadowed by raw features like Aircraft Mass.

To improve, incorporate advanced sampling like ADASYN for boundary-focused synthetics (potential 10-20% precision gain), scale to the full ~300K dataset for better generalization, and ensemble high-recall boosters with high-precision RF via VotingClassifier to balance FP/FN. Threshold tuning (e.g., >0.5 for positives) could further minimize alarms, while external data (e.g., bird migration APIs) refines proxies. Overall, the project underscores the challenges of imbalanced safety prediction but validates boosting for practical applications in aviation and agricultural drones, where enhanced models could prevent strikes during low-altitude operations. Future work should deploy with real-time monitoring for adaptive risk mitigation.