Final Project Report

Predicting Bird Strikes

Aasim Shaik - 02126256

CONTENTS

Introduction

1.1 Problem Statement:

A bird strike is strictly defined as a collision between a bird and an aircraft which is in flight or on a take-off or landing roll. Bird Strike is common and can be a significant threat to aircraft safety. For smaller aircraft, significant damage may be caused to the aircraft structure and all aircraft, especially jet-engine ones, are vulnerable to the loos of thrust which can follow the ingestion of birds into engine air intakes. This has resulted in several fatal accidents.

The main use of predicting bird Strikes include:

1) Mitigating safety Risks.
2) Reducing Aircraft Damage.
3) Improving Aviation safety Regulations.
4) Preserving Human Lives.

1.2 About Dataset:

The Dataset is collected from FAA (Federal Aviation Administration) during 2000 -2011. That documents all reported wildlife collisions with aircrafts within the USA.

It contains more than 200,000 bird strike incidents from 1990 to 2022. Each example has 25 attributes including- aircraft components affected, height of strike, airport details, damage level, bird/wildlife species, pilot awareness indicator, incident date, latitude, longitude, runway etc.

For this project, the dataset was preprocessed to extract 10 input features that were predictive for the output variable.

1.3 Feature Selection:

Output Feature: The output feature is binary, indicating whether a Bird Strike caused damage to the aircraft or not (0 indicate no damage and 1 indicate caused damage to aircraft).

Input Feature: The Input features are

- Engines: Number of engines involved in the aircraft.
- Flight Phase: Phase of flight during which the bird strike occurred.
- Feet above Ground: Altitude of the aircraft at the time of the bird Strike.
- Sky Conditions
- Pilot warned of birds: Whether the pilot was warned of the presence of birds.
- Birds Struck: Number of bird's s involved in the strike.
- Remains of wildlife collected: Whether the remains of wildlife were collected for analysis.

1.3 Purpose of Analysis:

The primary objective of analyzing this dataset is to develop a predictive model for bird strikes. By understanding the factors contributing to bird strikes, we aim to enhance aviation safety, reduce aircraft damage, and contribute to the development of effective preventive measures.

Data Collection

2.1 Labeling:

- **Target variable:** The target variable in this dataset is “Damage”, which serves as the label for each bird strike incident. The goal is to predict whether a given incident resulted in damage to the aircraft or not.
- **Binary Classification:** The labeling process involves creating a binary classification task where each incident is categorized into one of two classes:
    1) Class 0 (Negative class): The incident did not cause damage.
    2) Class 1(Positive class): The incident caused damage.

2.2 Balancing the Dataset:

- **Class imbalance:** In the context of bird strike incidents, there might be an imbalance between the number of incidents causing damage and those without damage.
- **Techniques for Balancing:** Oversampling the minority class: Duplicate instances of the minority class are added to the dataset, increasing its representation.

2.3 Data Statistics:

Total Training Examples used in the dataset are: 17123
Total Testing Examples used in the dataset are: 4217
Total validation Examples used in the dataset are: 4218

```
In [517]: df = df[order]
```

```
In [518]: df.head()
```

Out[518]:

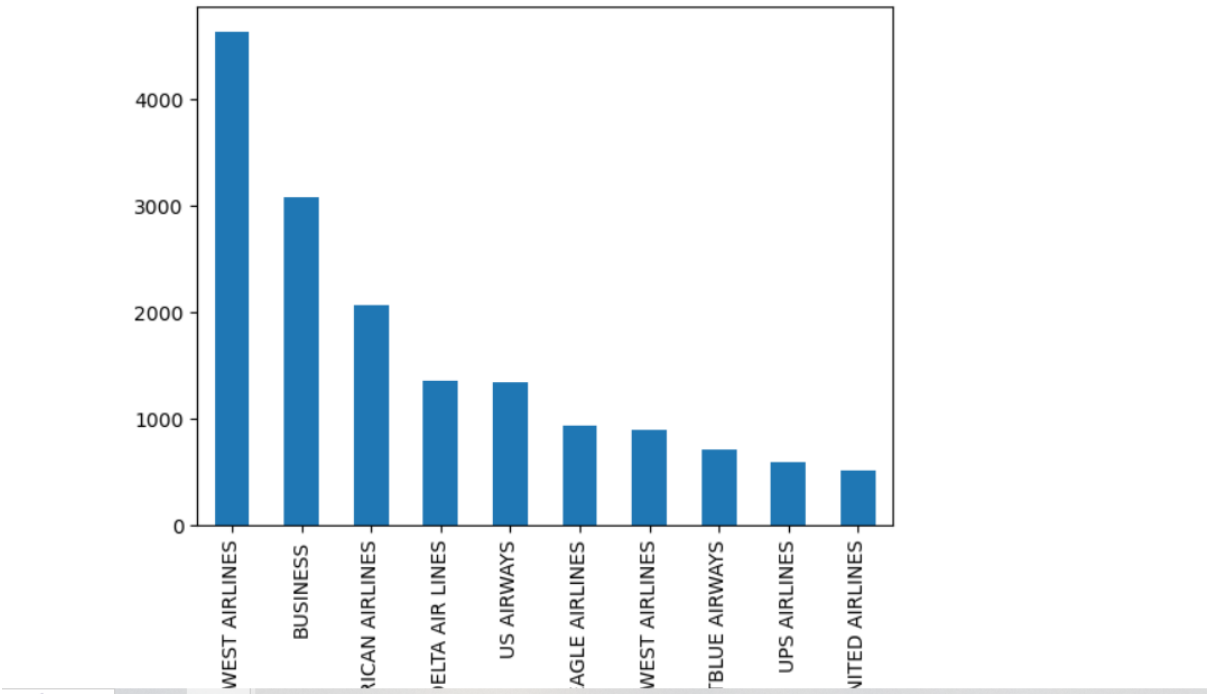
	Engines	Flight phase	Feet above ground	Sky Conditions	Pilot warned of birds	Birds Struck	Wildlife size	Remains of wildlife collected	Damage
0	2	Climb	1,500	No Cloud	N	859	Medium	False	Caused damage
1	2	Landing Roll	0	Some Cloud	Y	424	Small	False	Caused damage
2	2	Approach	50	No Cloud	N	261	Small	False	No damage
3	2	Climb	50	Some Cloud	Y	806	Small	True	No damage
4	2	Approach	50	No Cloud	N	942	Small	False	No damage

Visualization

3.1 Top 10 Airline Encountered by bird strike:

```
[507]: df["Aircraft: Airline/Operator"].value_counts()[:10].plot(kind='bar')
```

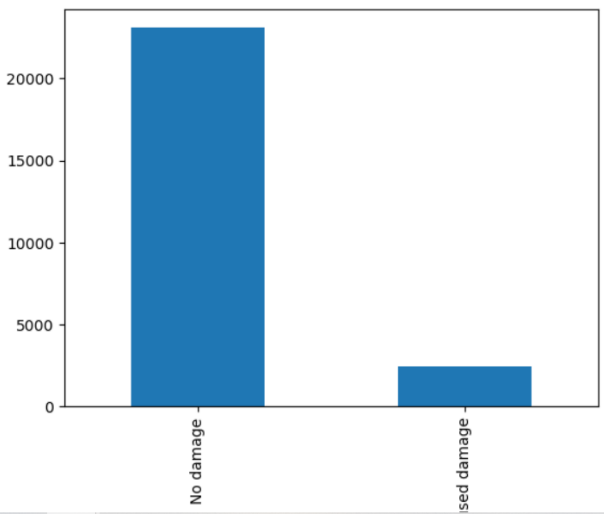
```
Out[507]: <Axes: >
```



3.2 Effect of Bird Strikes:

```
In [508]: df['Effect: Indicated Damage'].value_counts().plot(kind='bar')
```

```
Out[508]: <Axes: >
```



Data Preprocessing

4.1 Handling Missing values:

1) Identifying Missing values:

- **Observation:** Start by identifying missing values in the dataset. Common representations include Nan (Not a Number) for numerical data and special codes (e.g., “NA” or “Missing”) for categorical data.
- Replaced all the missing values of numerical columns with median of the data.
- Replaced all the missing values of categorical data with mode of the data.

4.2 Encoding:

Encoding is the process of converting categorical data into a numerical format that can be used for machine learning algorithms. Many machine learning models require numerical inputs, and encoding helps represent categorical variables in a way that these models can understand. In this machine learning code, encoding may involve converting categorical variables like "Flight phase" or "Sky Conditions" into numerical representations.

- Label Encoding: Assigns a unique numerical label to each category. Useful for ordinal categorical data.

4.3 Feature Scaling:

Feature scaling is a pre-processing technique that standardizes or normalizes the range of independent variables or features of a dataset. It is an essential step in many machine learning algorithms, especially those that rely on distance-based metrics or optimization algorithms. The goal of feature scaling is to ensure that all features contribute equally to the model's performance, preventing certain features from dominating the learning process. Two common methods for feature scaling are Min-Max Scaling and Standardization (Z-score normalization).

- **Min- Max Scaling:**
  Scales the data to a fixed range, usually between 0 and 1.

Logistic Regression

5.1 Logistic Regression Model: is a statistical method and a type of regression analysis used for predicting the probability of an outcome. Despite its name, logistic regression is used for classification tasks, especially binary classification where the outcome variable is categorical and has two classes (e.g., yes/no, 1/0).

- Logistic regression is widely used when the outcome variable is binary.
- The cost function is derived from maximum likelihood estimation.

5.2 Data Splitting:

The dataset was divided into training, validation, and test sets. The training set comprises 60% of the data,
The validation set 20%, and the test set 20%. This split ensures a robust evaluation of the model's performance.

5.3 Model Evaluation:

The model's performance was evaluated on the validation set using accuracy and confusion matrix metrics.

In [568]:
```
# Model evaluation
from sklearn.metrics import accuracy_score
# Calculate the accuracy score
accuracy = accuracy_score(y_val, y_val_pred)

# Display the accuracy score
print(f"Accuracy Score: {accuracy:.4f}")
```

```
Accuracy Score: 0.8981
```

In [569]:
```
from sklearn.metrics import classification_report

# Generate the classification report
report = classification_report(y_val, y_val_pred)

# Display the classification report
print("Classification Report:\n", report)
```

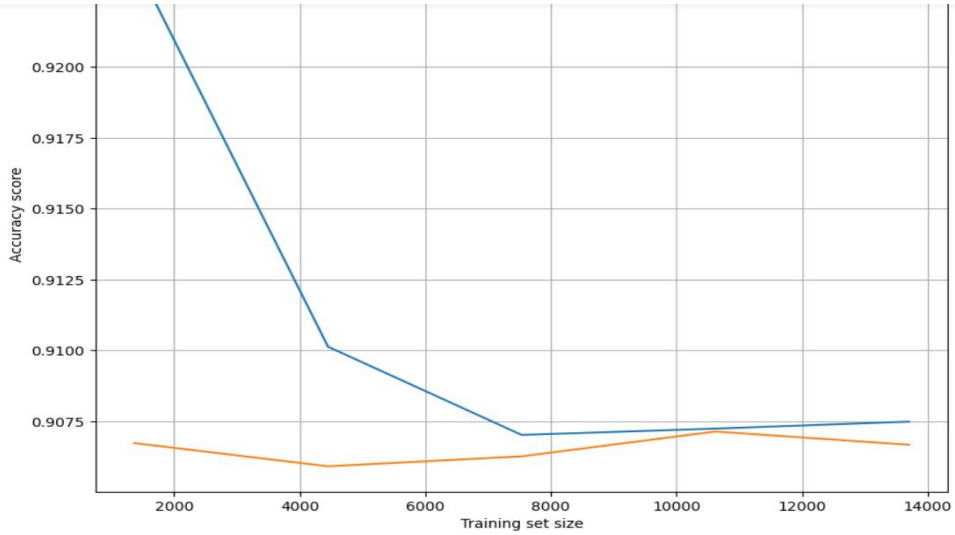
```
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.99      0.95      3789
           1       0.49      0.08      0.14       429

    accuracy                           0.90      4218
   macro avg       0.70      0.54      0.54      4218
weighted avg       0.86      0.90      0.86      4218
```

5.4 Precision, Recall and F1-score:

```
# Display the results
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-Score: {f1:.4f}")
```

```
Precision: 0.4928
Recall: 0.0793
F1-Score: 0.1365
```

5.5 Learning Curve:



The model is over fitting because, the training score is significantly higher than cross validation score across all training set sizes. This indicates that model fits the training data very well but generalizes poorly on unseen data.

5.6 Solution for over fit:

- Get more Training data.
- Try regularization techniques like L1/l2 regularization to reduce over fitting.
- Try simpler models with fewer parameters to reduce over fitting.
- Tune hyper parameters through cross validation to find the optimal model complexity.

Regularization and Tuning

6.1 Problem Statement:

Implemented a pipeline for logistic regression, including standardization using Standard Scalar logistic regression with L2 regularization using Logistic Regression, and hyper parameter tuning using GridSearchCV.

1) Pipeline Construction:

- A pipeline was constructed to streamline the preprocessing and modeling steps.
- Components of the pipeline are:
- Standard Scalar: Standardizes numerical features to ensure consistent scales.
- Logistic Regression with L2 Regularization: Implements logistic regression with L2 regularization to prevent over fitting.
- The 'liblinear' solver was chosen for logistic regression.

2) Hyper parameter Tuning:

- **Grid Search:** Utilized 'GridSearchCV' to perform an exhaustive search over a specified parameter grid.
- Cross-validated the pipeline using 5-fold cross-validation (**CV=5**).
- The hyper parameter of interest was the regularization strength (C) for logistic regression

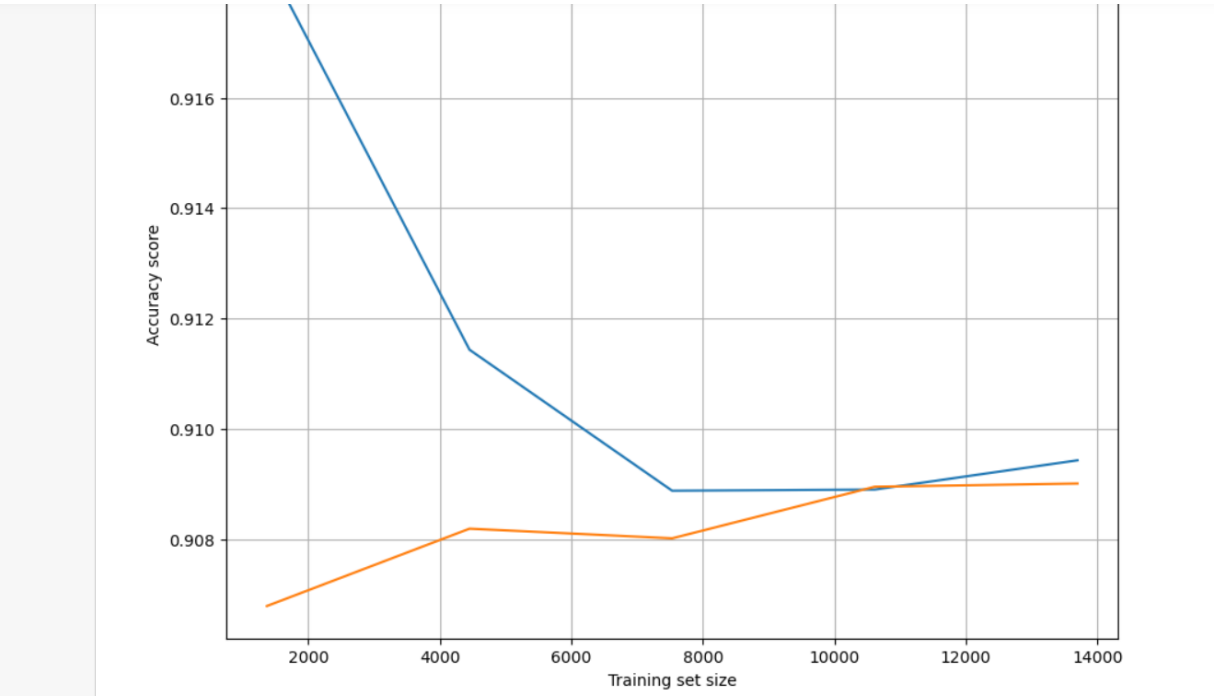
6.2 Test and Validation matrices:

```
# Make predictions on the validation set
y_val_pred = best_model.predict(X_val)

# Evaluate the model on the validation set
accuracy_val = accuracy_score(y_val, y_val_pred)
precision_val = precision_score(y_val, y_val_pred)
recall_val = recall_score(y_val, y_val_pred)
f1_val = f1_score(y_val, y_val_pred)

# Display the evaluation metrics
print("\nValidation Set Results:")
print(f"Accuracy: {accuracy_val:.4f}")
print(f"Precision: {precision_val:.4f}")
print(f"Recall: {recall_val:.4f}")
print(f"F1-Score: {f1_val:.4f}")
```

```
Best Hyperparameters: {'classifier__C': 0.01}

Validation Set Results:
Accuracy: 0.9030
Precision: 0.6613
Recall: 0.0956
F1-Score: 0.1670
```

The accuracy of the model has been increased after performing Regularization and hyper parameter tuning.

6.3 Learning Curve:



This is a good fit model because:

The gap between the curves is less and both training and validation has good accuracy.

6.4 Comparative Analysis:

```
# Calculate the number of incorrect predictions
incorrect_preds = (y_val != y_val_pred).sum()

# Display the count of incorrect predictions
print(f"Number of incorrect predictions: {incorrect_preds}")
```

```
Number of incorrect predictions: 430
```

- Logistic Regression and Updated Logistic Regression after Regularization and Hyper parameter Tuning

```
# Calculate the number of incorrect predictions
incorrect_predictions_tuned = (y_val != y_val_pred_tuned).sum()

# Calculate accuracy on the validation set using the tuned model
accuracy_val_tuned = accuracy_score(y_val, y_val_pred_tuned)

# Display the results
print(f"Number of incorrect predictions (after tuning): {incorrect_predictions_tuned}")
print(f"Accuracy on the validation set (after tuning): {accuracy_val_tuned:.4f}")
```

```
Number of incorrect predictions (after tuning): 409
Accuracy on the validation set (after tuning): 0.9030
```

The model has been improved after performing Regularization and Hyperparameter tuning

6.5 Ensemble Model:

- Random Forest Ensemble: A Random Forest ensemble model was chosen for its ability to handle complex relationships in data and provide robust predictions.
- Model Configuration: Number of Trees = 100 and Random state 42(for reproducibility).
- Training The Model: The Random Forest model was trained on the training set (xtrain, ytrain).
- Making Predictions on the Validation Set:
  The trained Random Forest model was used to make predictions on the validation set (Xval).

Support Vector Machine

7.1 SVM: An SVM model with a linear kernel was selected for its effectiveness in handling linearly separable data.

7.2 Training the Model: The SVM model was trained on the training set (Xtrain, Ytrain)

SVMs aim to find the hyper plane that best separates the data into different classes.

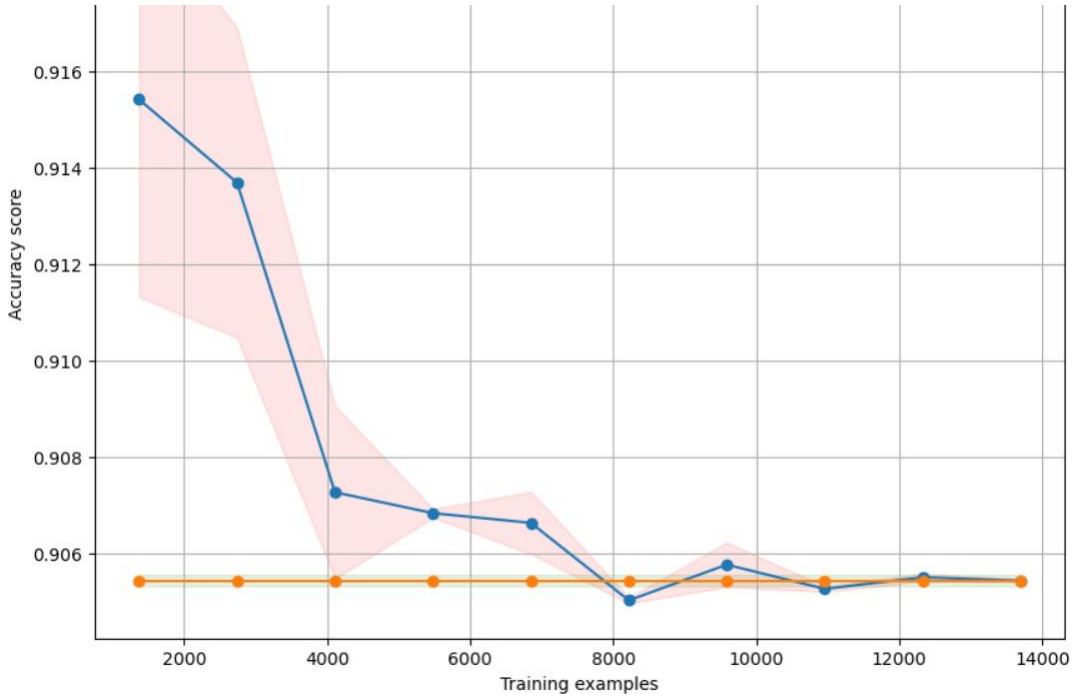
7.3 Model Evaluation:

- Accuracy: Computed the accuracy of the SVM model on the validation set.

```
# Obtain predictions on the validation set using the SVM model
y_val_pred_svm = svm_model.predict(X_val)

# Calculate the number of incorrect predictions
incorrect_predictions_svm = (y_val != y_val_pred_svm).sum()

# Calculate accuracy on the validation set using the SVM model
accuracy_val_svm = accuracy_score(y_val, y_val_pred_svm)

# Display the results
print(f"Number of incorrect predictions (SVM): {incorrect_predictions_svm
print(f"Accuracy on the validation set (SVM): {accuracy_val_svm:.4f}")
```

```
Number of incorrect predictions (SVM): 429
Accuracy on the validation set (SVM): 0.8983
```

7.4 Learning Curve:

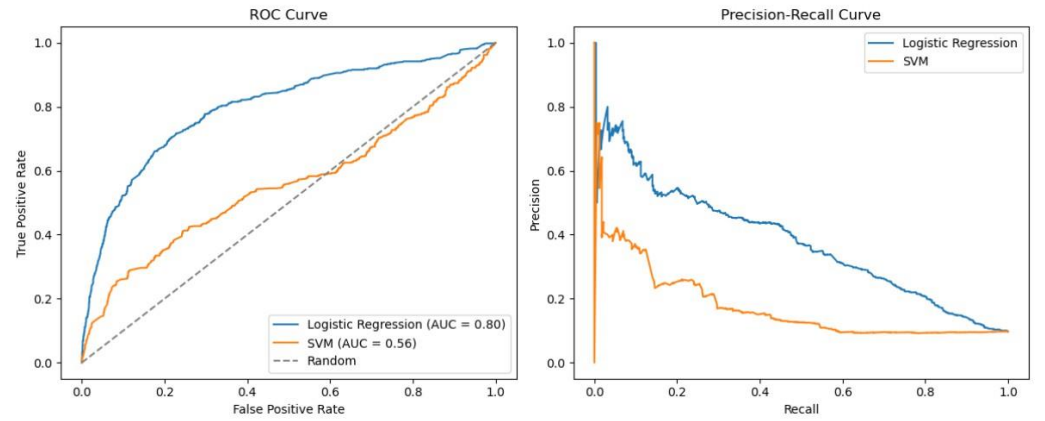


Conclusion

```
n_iter_i = _check_optimize_result(

Logistic Regression:
Accuracy: 0.9063
Precision: 0.6250
Recall: 0.1002
F1 Score: 0.1727
ROC AUC: 0.5468

SVM:
Accuracy: 0.9024
Precision: 0.0000
Recall: 0.0000
F1 Score: 0.0000
ROC AUC: 0.5000

Random Forest:
Accuracy: 0.9053
Precision: 0.5366
Recall: 0.2204
F1 Score: 0.3125
ROC AUC: 0.5999
```

Hence the best model is Logistic Regression with performing Regularization and Hyper parameter Tuning.

```
plt.tight_layout()
plt.show()
```



This project provides a comprehensive analysis of machine learning models in predicting bird strike incidents. Each model demonstrated strengths, and the choice of the final model should align with the project's specific requirements and constraints. Continuous improvement and adaptation will be key in maintaining the model's effectiveness over time. The insights gained from this project

contribute to aviation safety efforts by providing a predictive tool to anticipate and mitigate the impact of bird strikes on aircraft.